

ترجمة سنة 2008
لمراسلة المترجم: ashrafkhalaf@gmail.com

بسم الله الرحمن الرحيم

الحمد لله الذي بنعمته تتم الصالحات ، والصلاة والسلام على النبي المبعوث بالرحمات، وعلى آله وصحبه ذوي المكرمات، وأمهات المؤمنين الطيبات الطاهرات، وسلم تسليما كثيرا ، وبعد ، ،

فهذا أول كتاب ينطق باللسان العربي ، في مجال بناء وتثبيت وتحديث وترقية وصيانة نواة لينكس ، ويقدم للقارئ العربي معلومات وفيرة عن طريقة القيام بذلك الأمر الذي كان يعد من الأمور الغامضة على الكثيرين من مستخدمي نظام لينكس الرائع ، ويُشعر المستخدم للنظام بقيمة النظام الذي يستخدمه، وكيفية عمله ، ويفتح له آفاق الإبداع والمشاركة في تطوير النواة إن كان يمتلك الأدوات اللازمة لذلك ،

وسبب اختياري لهذا الموضوع هو فقر المكتبة العربية إن لم يكن خلوها من هذه النوعية من الكتب والتي تنقل المستخدمين والمطورين خطوات كبيرة للأمام موفرة عليهم عناء الترجمة والفهم للنص المكتوب بغير لسانهم

والكتاب ترجمة لأحد أشهر الكتب في نواة لينكس

Linux Kernel in a Nutshell

لأحد كبار مطوري النواة والهاكر الكبير

Greg Kroah-Hartman

وقد قمت بحمد الله بترجمة الكتاب باللسان العربي إضافة إلى وضع الكثير من الهوامش المفيدة عن الكثير من المصطلحات والأجهزة والأسماء الواردة في الكتاب والتي يستفيد القارئ من التعرف عليها مما يزيد من قيمة الكتاب

العلمية ، وقد وضعت لجميع الهوامش أرقاما بينما أشير إلى الهوامش الخاصة بالكتاب الأصل - وهي قليلة - من خلال نجمة(*) أو نجمتين (**)

وفي النهاية ، برجاء ممن لديه أية مقترحات أوملاحظات على أي جزء من العمل سواء في الترجمة أو التنسيق، أو يريد التعاون معنا في هذا الإطار أن يتواصل معنا بمراسلتنا على البريد الإلكتروني

ashrafkhalaf@gmail.com

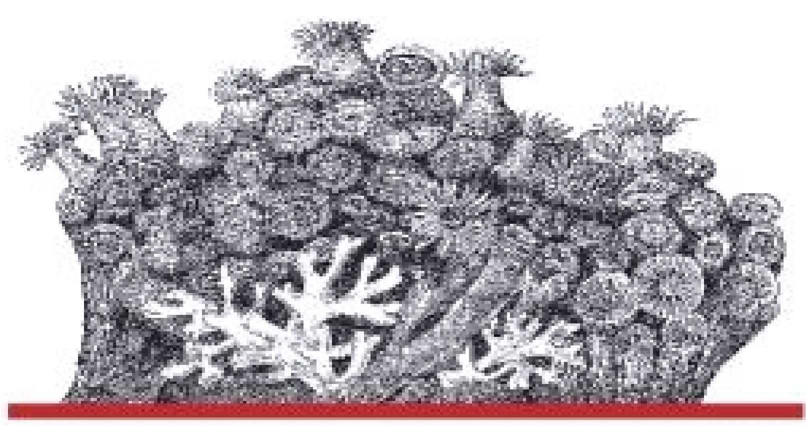
shararf969@yahoo.com

والله أسأل أن ينفع به القارئ الكريم ، ولا تنسوا الدعاء لنا ولوالدينا بالرحمة والرضوان من الله السميع العليم

أشرف علي خلف

مصر - الإسكندرية

19/10/2008



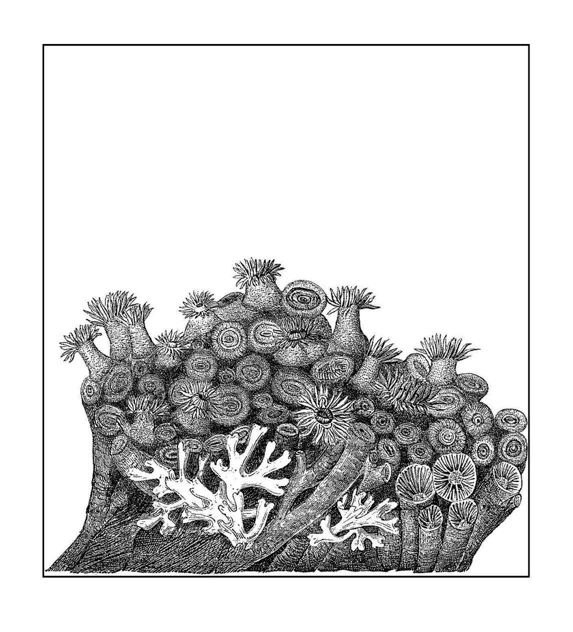
LINUX KERNEL

IN A NUTSHELL

A Desktop Quick Reference

O'REILLY®

Greg Kroah-Hartman



الافتتاحية

عندما خطر لي موضوع هذا الكتاب لأول وهلة
صرفت النظر عنه وذلك مثل كل الأمور التي
تم تغطيتها بوفرة بالفعل من قبل الوثائق الخاصة بنواة لينكس.
فمن المؤكد أن شخصا ما قام بالفعل بالكتابة عن كل الأساسيات اللازمة لبناء
وتركيب، وتعديل نواة لينكس ، وذلك يبدو مهمة بسيطة بالنسبة لي *.
ولكن بعد التنقيب في مستندات Howtos المختلفة، ومن خلال الوثائق في دليل
نواة لينكس، وصلت إلى استنتاج انه لا يوجد فيها مكان واحد يمكن أن تجتمع فيه كل
هذه المعلومات. ويمكن أن أستقيها عن طريق عدد قليل من المراجع والملفات من هنا
أو هناك ، وعدد قليل من المواقع التي عفا عليها الزمن على شبكة الإنترنت ، ولكن
هذا لم يكن مقبولا لأي شخص لا يعرف بالضبط أين يبحث في المقام الأول.
ولذا تم تأليف هذا الكتاب بهدف توحيد جميع المعلومات الموجودة بالفعل والمتناثرة
في ثنايا شبكة الإنترنت عن بناء نواة لينكس ، بالإضافة إلى الكثير من المعلومات
المفيدة والجديدة التي لم تكن مكتوبة فيما سبق في أي مكان ولكن تم اكتسابها من
التجربة والخطأ خلال سنوات من عملي في تطوير نواة لينكس .
والغرض الخفي لي من تأليف هذا الكتاب هو جذب المزيد من الناس إلى حظيرة
تطوير نواة لينكس .إن عملية بناء نواة مخصصة لجهازك تعد واحدة من المهام
الأساسية اللازمة لتصبح مطورا لنواة لينكس. وإن المزيد من الأشخاص الذين
يحاولون القيام بذلك وإدراك أنه لا يوجد سحر حقيقي وراء جميع عمليات النواة،
والمزيد من الناس سيكونون مستعدين للتقدم بيد المساعدة في صناعة النواة وجعلها
في أفضل حال يمكن أن تكون عليه.

---

* أنا مطور محترف لنواة لينكس ، ولذلك قد تبدو الأمور بدائية وبسيطة بالنسبة لي ،في حين أنها في أوقات أخرى تبدو غير مفهومة لمعظم الناس ،مثل أفراد عائلتي الذين طالما ذكروني بذلك الأمر.

لمن هذا الكتاب :

هذا الكتاب يهدف إلى تغطية كل شيء لازم لمعرفة الطريقة الصحيحة لبناء، وتعديل، وتثبيت نواة لينكس. ليس هناك حاجة لخبرة بالبرمجة لفهم واستخدام هذا الكتاب . فقط بعضا من المعرفة بكيفية استخدام لينكس وبعض الأساسيات عن استخدام سطر الأوامر كما هو متوقع من القارئ .

هذا الكتاب لا يهدف إلى الخوض في الجوانب البرمجية في نواة لينكس،فهناك الكثير من الكتب الجيدة والمدرجة في قائمة المراجع التي تغطي بالفعل هذا الموضوع .

كيفية تنظيم هذا الكتاب :

هذا الكتاب ينقسم إلى أربعة أجزاء

الجزء الأول : بناء النواة

ويشمل الفصول من 1 إلى 6 والتي تغطي كل شيء تحتاج لمعرفته عن جلب وبناء وتثبيت وترقية نواة لينكس وسواء زاد ذلك أو قل سيكون بطريقة خطوة بخطوة.

الفصل الأول : مقدمة

يشرح هذا الفصل متى ولماذا نرغب في بناء النواة.

الفصل الثاني : متطلبات بناء واستخدام النواة

يغطي هذا الفصل البرامج المختلفة والأدوات اللازمة للبناء الصحيح للنواة .ويغطي كذلك العديد من البرامج المختلفة ذات الصلة الوثيقة بالنواة،وكيف تنتقي إصدارات البرامج وأين يمكنك العثور عليها.

الفصل الثالث : الحصول على الملف المصدري للنواة

هذا الفصل يناقش كيف أن إصدارات نواة لينكس يرتبط بعضها ببعض،وأين يمكنك الحصول على شفرة المصدر الخاصة بنواة لينكس ،وكيفية تحميلها بشكل صحيح.

الفصل الرابع : تهيئة وبناء النواة

هذا الفصل يشرح كيفية تهيئة وبناء نواة لينكس بشكل سليم .

الفصل الخامس : التثبيت والإقلاع من النواة

هذا الفصل يريك كيف تثبت النواة التي تم بناءها بشكل صحيح، والإقلاع داخل هذا الإصدار من النواة.

الفصل السادس : ترقية النواة

هذا الفصل يشرح كيفية ترقية نواة تم بناؤها مسبقا إلى إصدار أحدث بدون الاضطرار للبدء من الصفر.

الجزء الثاني التعديلات الرئيسية - Major Customizations

ويتألف من الفصلين 7 و 8 ، التي تصف كيفية تهيئة النواة بشكل صحيح على أساس ألأجهزة الموجودة في النظام ، ويزودنا بعدد من "الوصفات " الشائعة لهذه الإعدادات ؟

الفصل السابع ، تخصيص النواة

ويناقش هذا الفصل كيفية تخصيص النواة للأجهزة والعتاد الموجود على النظام. ويمر على مجموعة متنوعة من الطرق المختلفة لتحديد الخيارات التي يجب تحديدها ويمدنا ببعض السكربتات البسيطة للمساعدة في هذه المهمة.

الفصل الثامن : وصفات تهيئة النواة

هذا الفصل يوضح كيفية تهيئة النواة لمجموعة متنوعة من الحالات الشائعة .

الجزء الثالث مرجع لأوامر وخيارات النواة

ويشمل الفصول من 9 إلى 11. هذه الفصول تقدم لنا مرجعا للخيارات المختلفة لسطر الأوامر الخاص بالنواة ، وخيارات بناء النواة وقليل من الخيارات المختلفة لتهيئة النواة.

الفصل التاسع : مرجع لمعاملات أوامر إقلاع النواة

هذا الفصل يهتم بجميع التفاصيل المختلفة لخيارات سطر الأوامر التي يمكن تمريرها إلى النواة ، وماذا تقوم به هذه الخيارات المختلفة.

الفصل العاشر : مرجع بأوامر بناء النواة

يصف هذا الفصل سطر الأوامر مختلف الخيارات المتاحة عند بناء النواة وكيفية استخدامها.

الفصل الحادي عشر : مرجع بخيارات تهيئة النواة

هذا الفصل يلقي الضوء على عدد قليل من الخيارات الأكثر شعبية وأهمية في تهيئة نواة لينكس .

الجــزء الرابـــع معلومات إضافية

الملحق A أدوات مساعدة

هذا القسم يقدم لك عددا من أجود الأدوات وأكثرها نفعا والتي يحتاج إليها كل شخص لخوض غمار أحدث إصدارة من نواة لينكس ويلزمه استخدامها.

الملحق B: فهرس المراجع

هذا القسم يقدم لك قائمة من المراجع النافعة والتي يمكنك استخدامها في تتبع المزيد من المعلومات في بنائك للنواة الخاصة بك.

رخصة الكتاب على شبكة الإنترنت

هذا الكتاب متاح مجانا تحت رخصة الإبداع العامة غير التجارية والمشاركة بالمثل الإصدار رقم 2.5.

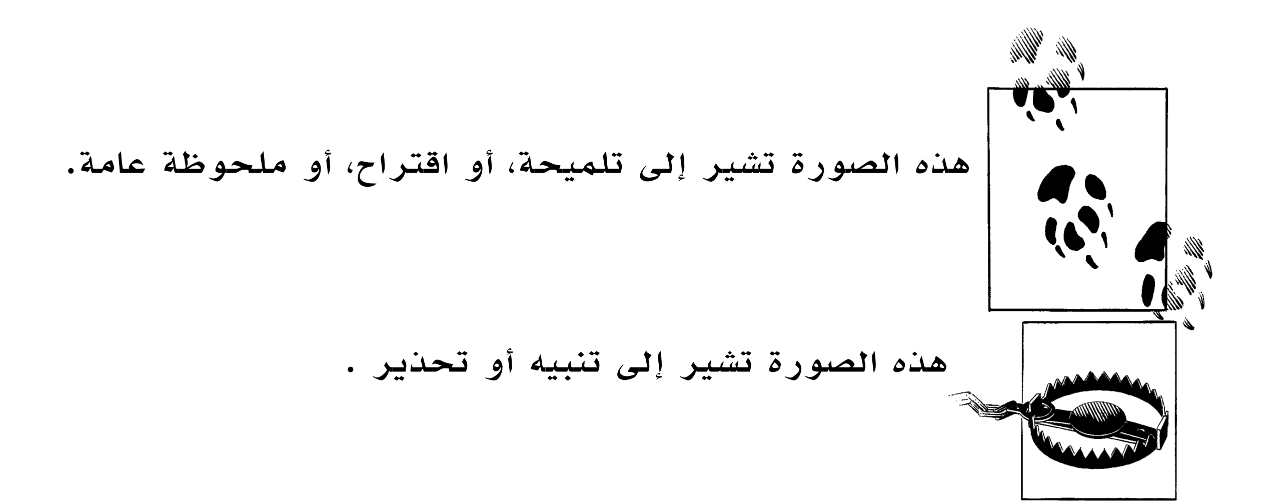
هذه الرخصة يمكنك الاطلاع عليها بالكامل في
http://creativecommons.org/licenses/by-sa/2.5/

والكتاب كاملا متاح أيضا على شبكة الإنترنت في
http://www.kroah.com/lkn

أمور متفق عليها في هذا الكتاب

الخط المائل **: يشير إلى البرامج ، الأدوات ، الأوامر ، خيارات الأوامر، اسماء حزم التوزيعات، الملفات ، الأدلة ، أسماء المستخدمين ، والمضيفين، وكذلك يشير إلى تسمية لم نستخدمها في السابق.**

#, $ تستخدم لبعض الأمثلة مثل محث المستخدم الجذر # والمستخدم العادي $ ، تحت صدفة الباش .



هذه الصورة تشير إلى تلميحة، أو اقتراح، أو ملحوظة عامة.

هذه الصورة تشير إلى تنبيه أو تحذير .

كيفية التواصل معنا :

لقد قمنا بالفحص والتحقق من جميع المعلومات الواردة في هذا الكتاب على قدر ما أوتينا من قوة ، ولكنك قد تجد ان بعض المواصفات قد تغيرت (أو حتى التي أخطأنا فيها!). يرجى إعلامنا عن أي أخطاء تعثر عليها، وكذلك المقترحات الخاصة بك من أجل الطبعات المقبلة للكاتب :

O’Reilly Media, Inc.

1005 Gravenstein Highway North

Sebastopol, CA 95472

800-998-9938 (in the United States or Canada)

707-829-0515 (international/local)

707-829-0104 (fax)

يمكنك أيضا أن ترسل إلينا رسائل إلكترونية. ومن أجل أن توضع على القائمة البريدية أو طلب قائمة أرسل بريدا إلكترونيا إلى : info@oreilly.com
وللسؤال أسئلة تقنية او التعليق على الكتاب ، أرسل بريدا إلكترونيا الى :
bookquestions@oreilly.com

لدينا موقع على الشبكه العالمية للكتاب ، حيث سنقوم بوضع قائمة للأمثلة ، والأخطاء ، وأية خطط
للطبعات المستقبلية. يمكنك الوصول الى هذه الصفحة في :

http://www.oreilly.com/catalog/9780596100797

اعترافات:

أحب أن أبدأ اولا بشكر زوجتي الرائعة شانون، وطفلتي الجميلتين مادلين وجريفين على تفهمهم وصبرهم خلال فترة تأليفي لهذا الكتاب . ولم يكن ممكنا لهذا الكتاب - بدون دعمهم وصبرهم - أن يكتب له الاكتمال أبدا. وشكر خاص موجه لشانون لأنها أخذت بيدي إلى مجال تطوير نواة لينكس فى المقام الاول. وبدون اجتهادها، لكنت الآن ما زلت أقوم بعمل بعض الأعمال البرمجية التافهة والكاسدة.ولم أكن لأكتشف ذلك المجتمع العظيم الذي أعمل من خلاله.

المحرر الخاص بي آندي أورام، يعتبر بمثابة القوة الدافعه التي تقف وراء هذا الكتاب ،وقد صاغه بشكل يجمع بين قابليته للقراءة والغني بالمعلومات. ومهاراته في التحرير وصبره قد بلغت غايتها النهائية، وكانت ذات أثر كبير في إنشاء وإنجاز هذا الكتاب.

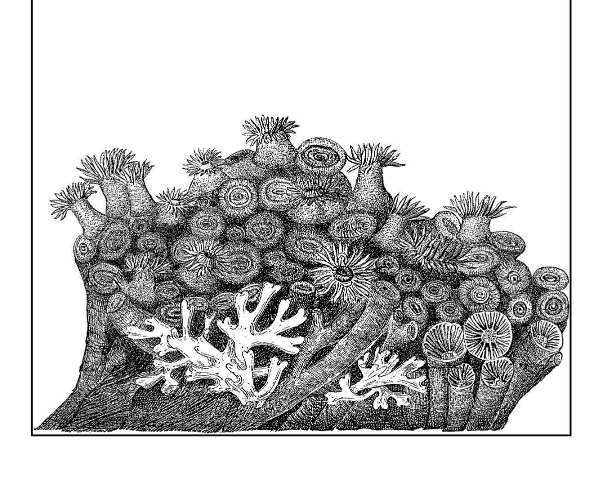
وشكر كبيرأيضا للمحرر الأصلي لهذا الكتاب ، ديفيد بريكنر، لإعطائه لي فرصة العمل في هذا المشروع والإيمان بقدرتي على إنجازه منذ البداية، على الرغم من أن النسخة الاولى كانت في اكثر من 1000 صفحة.

المراجعون الفنيون لهذا الكتاب كانوا مذهلين ، حيث إنهم التقطوا جميع الأخطاء الوفيرة واكتشفوا ما حدث من السهو والثغرات التي يتعين سدها.

وكان المراجعون (حسب الترتيب الأبجدي بالاسم الأول، وليس تبعا لمهاراتهم العظيمة) ، كريستيان مورجنر ، جولدن.ج. ريتشارد الثالث ، جين ديلفار، جير كوبرشتاين، مايكل بوينر ،ريك فان ريل ، وروبرت داي.

شكر خاص ل راندي دونلاب لخوضه في معاملات إقلاع النواة وتمشيطها بشكل أنيق وتقديم الملاحظات في هذا الفصل ، وكذلك الشكر ل كاي سييفرز الذي ساعد كثيرا في كل الفصل الخاص بتخصيص النواة ، وهو الذي زودنا بسكربت في نهاية نفس الفصل. وبدون مساعدته لنا بـ *sysfs* ومعارفه ، لم يكن ممكنا لهذا الفصل أن تتم كتابته .

وفي النهاية شكر خاص الى معلمي في اللغة الإنجليزيةفي الصف السادس ، والسيدة جرابر ، الذين علماني الكتابة التي كانت في بعض الأحيان مستحيلة الحدوث ، وعلموني الاستمتاع أثناء القيام بذلك. وبدون هذه البداية ، لم يكن ليتحقق أي شيء من هذا.



1



مقدمـــــــــــــــة

على الرغم من كبر قاعدة الشفرات في لينكس (أكثر من 7مليون سطر) إلا أنه يعتبر من أكثر نظم التشغيل مرونة التي تم إنشاؤها على الإطلاق ،فهو من الممكن أن يتحول أو ينشأ من خلاله أنظمة متنوعة لعمل أي شيء مثل وحدة التحكم الراديوي في الطائرات المروحية أو الهاتف الخلوي (المحمول) والغالبية العظمى من الحاسبات العملاقة- أو الخوادم -في العالم ......إلخ

ويتم ذلك عن طريق التعديل في النواة وفقا للبيئة المناسبة لك .

وإنه لمن المستحيل أن يتم عمل شيء يجمع بين سهولة وسرعة النواة الموجودة في داخل توزيعات لينكس

وهذا الكتاب سوف يبحث في كيفية بناء وتركيب النواة، ويزودنا ببعض التلميحات عن كيفية تفعيل الخيارات المحددة التي يحتمل أن نستخدمها في حالات مختلفة.

لا توجد نواة لينكس تمد مستخدميها بالضبط بكل ما يحتاجونه، ولكن التوزيعات الحديثة أصبحت ملائمة جدا وتحتوي على دعم لكافة الأجهزة المعروفة، بدءا ببطاقات الصوت وحتى موفرات الطاقة، ولكنك ستحتاج ببساطة إلى أشياء تختلف عن الأغلبية العظمى من المستخدمين (وكل التوزيعات تحاول تلبية احتياجات أغلب المستخدمين).

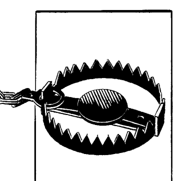
ربما يكون لديك عتاد مختلف عن الآخرين وعندما تخرج إحدى إصدارات النواة إلى الوجود فربما ترغب في استخدامها دون أن تنتظر توزيعة مبنية علي تلك النواة.

ولعدة أسباب، في بعض الأحيان ربما تريد أثناء عملك مع لينكس بناء النواة أو تعدل المعاملات في أحد الأنوية التي تعمل عليها.

وهذا الكتاب يعطيك المعلومات التي تحتاجها لفهم النواة من وجهة نظر مستخدم، ولتقوم بعمل أكثر التعديلات شيوعا عليها . وهناك أيضا بعض الأسباب الوجيهة

لحذف بعض المزايا من النواة خصوصاً إذا كنت تعمل ضمن مؤسسة إنتاجية صغيرة.
وعندما تبدأ بالتعديلات، فمن المفيد أن تفهم السمات الداخلية للنواة، وذلك يقع في خارج نطاق هذا الكتاب اللهم إلا بعض الملخصات الموجزة والتي تظهر مع خيارات معينة.
ملحق B الخاص بالكتاب يتضمن مرجعا لكتب أخرى، ومواد علمية يمكنها أن تعطيك مزيدا من الأساسيات عن الموضوع.

استخدام هذا الكتاب



لا تقم بأي إعدادات أو بناء للنواة بصلاحيات المستخدم الجذر. هذا التحذير في غاية الأهمية، وعليك أن تتذكره أثناء عملك بخطوات هذا الكتاب . كل شيء في هذا الكتاب، مثل : تحميل الملف المصدري للنواة من الإنترنت ، أو فك أرشفتها، أو تهيئتها، وبنائها ، كل ذلك يجب أن يتم وأنت مستخدم عادي على النظام .
فقط هناك أمران أو ثلاثة أوامر تحتاج لصلاحيات المستخدم الجذر (*root*).

قديماً كانت توجد بعض الأخطاء البرمجية (bugs) أثناء عملية بناء النواة يتولد عنها حذف بعض الملفات الخاصة في الدليل */dev* ، في حالة ما كان المستخدم يعمل بصلاحيات المستخدم الجذر أثناء عملية بناء النواة (*)
وكذلك هناك عواقب تنشأ بسهولة عند فك ضغط نواة لينكس بواسطة صلاحيات المستخدم الجذر، مثل بعض الملفات في الملف المصدري للنواة والتي لن تنتهي مع الصلاحيات المناسبة، وسوف تسبب الأخطاء في بناة النواة لاحقا .
وكذلك يجب ألا يوضع الملف المصدري أبدا في المجلد */usr/src/linux/*
حيث إن هذا المسار يحتوي على النواة الأصلية الذي تم بناء مكتبيات النظام عليها، وليس الكيرنل الجديد الذي اخترته أنت.

(*)هذا المسلك الخاطئ استغرق زمنا لإصلاحه ،كما أنه لم يكن أحد من مطوري النواة الأوائل يبني النواة وهو مستخدم جذر، لذلك لم يكن أحدهم يعاني من هذه الأخطاء (bugs). وقد ضاعت عدة أسابيع قبل أن يحددوا في النهاية أن عملية بناء النواة هي المشكلة.
عدد من مطوري النواة اعتقدوا بشكل شبيه بالسخرية أن هذه الأخطاء (bugs) إنما وجدت لتساعد على منع أي شخص من بناء النواة بصفة المستخدم الجذر ،ولكن هدأت العقول بعدها وتم إصلاح الأخطاء في بناء النظام .

على كل حال لا تقم بعمل أي تحديث للنواة في المسار /usr/src/ ،لكن قم بعمل ذلك في دليل المستخدم (home)الخاص بك فقط حيث لا يمكن حدوث أي ضرر للنظام.



2

متطلبات بناء واستخدام النواة

هذا الفصل يشرح البرامج التي تحتاجها لإعداد النواة وبنائها، وإقلاعها بنجاح.وإنها لفكرة ذكية أن تقوم بالاطلاع على التغيرات في الوثائق للتحقق من رقم الإصدارة المحددة التي عليك أن تستخدمها في كل أداة تم شرحها في هذا الفصل.

هذا الفصل يرتكز على نواة بإصدارة رقم 2.6.18 ، وسوف يبين لك أرقام إصدارات الأدوات التي تعمل مع النواة . فإذا كنت تستخدم نواة مختلفة، فرجاء قم بالتحقق من أن لديك الإصدارات اللازمة المحددة في هذا الملف، وإلا فإن بعض الأمور لن تعمل بشكل صحيح، ومن ثم فإنه سيكون من الصعب تحديد مكمن الخطأ .

أدوات بناء النواة :

أغلب توزيعات لينكس تعرض عليك خيارا عند التنصيب، عبارة عن مجموعة من الحزم تسمى kernel hacking packages، فلو كانت توزيعتك تعرض عليك هذا الخيار فإنه من السهل عليك تثبيتها بدلا من محاولة تثبيتها بعد ذلك كبرامج منفردة لازمة لأداء هذه المهمة.

يوجد ثلاث حزم فقط تحتاج إليها لتتم عملية بناء النواة بنجاح:

1. المترجم (compiler)
2. الرابط (linker)
3. الأداة make

هذا الفصل يوضح محتوى كل حزمة مما سبق.

المترجم (compiler)

لقد تم كتابة نواة لينكس بلغة C مع قدر قليل من لغة التجميع في بعض المواضع. ولبناء النواة يجب أن نستخدم المترجم gcc C compiler .

أغلب توزيعات لينكس تتضمن حزمة gcc مثبتة مع النظام .
وإذا كنت ترغب في تحميل هذه الحزمة وتثبيتها بنفسك يمكنك الحصول عليها من هنا http://gcc.gnu.org .
وبداية من الإصدار رقم 2.6.18 من النواة ، فإن النسخة رقم 3.2 من gcc تعتبر أقدم نسخة تعمل مع النواة بشكل سليم. كن حذرا حيث إن اغلب نسخ gcc الحالية - الغالب - ليست خيارا جيدا دائما. حيث إن بعض حزم gcc الحديثة لا تقوم ببناء النواة بشكل سليم، لذلك لو لم تكن ترغب في المساعدة في تصحيح أو إزالة أخطاء المترجم ، فلا ننصح بتجربتها.
وللتحقق من ماهية نسخة gcc على نظامك اكتب هذا الأمر

```
$ gcc --version
```

الرابط Linker

إن مترجم لغة سي، gcc، لا يقوم وحده بعمل كل شيء في عملية الترجمة. فهو يحتاج إلى مجموعة إضافية من الأدوات تعرف بـ *binutils* لعمل الربط والتجميع بين الملفات المصدرية.وتحتوي حزمة *binutils* أيضا على بعض الأدوات المفيدة التي يمكنها معالجة الملفات المستهدفة بعدة طرق مختلفة ومفيدة مثل عرض مكونات مكتبة.
binutils يمكن عادة أن توجد في حزمة داخل التوزيعة تدعى (بلا اندهاش) .
binutils ، وإذا كنت ترغب في تنزيل وتثبيت الحزمة بنفسك يمكن أن تجدها هنا *http://www.gnu.org/software/*binutils
وبدءا من الإصدارة 2.6.18 من النواة، فإن إصدارة 2.12 من *binutils* تعتبر أقدم نسخة مستقرة وناجحة لربط النواة.
وللتحقق من رقم إصدارة *binutils* في نظامك اكتب الأمر التالي

```
$ ld -v
```

make

تعتبر *make* *أداة* تنتقل بداخل شجرة الملفات المصدرية لتحدد أيا من الملفات لازمة لعملية الترجمة، ومن ثم تقوم باستدعاء المترجم ، وأدوات البناء الأخرى لعمل بناء للنواة. ويحتاج الكيرنل لأحد إصدارات *make* التابعة لمشروع GNU والذي يوجد عادة في حزمة تسمى *make* *داخل توزيعتك؟*
إذا كنت ترغب في تنزيل حزمة *make* *وتثبيتها* بنفسك يمكنك العثور عليها في الموقع http://www.gnu.org/software/make
وبدءا من الإصدارة 2.6.18 النواة ، فإن إصدارة 3.97.1 من *make* *تعتبر* أقدم نسخة يمكنها بناء النواة بشكل سليم.

ومن الموصى به أن تقوم بتثبيت آخر إصدارة مستقرة من الحزمة *make*، ؛حيث *إن* النسخ الحديثة معروفة بأنها تعمل بشكل أسرع في بناء الملفات.
وللتحقق من رقم إصدارة *make* في نظامك اكتب الأمر التالي

```
$ make –version
```

أدوات لاستخدام النواة

بينما تعمل إحدى إصدارات النواة ،فإنها عادة لا تؤثر على أي تطبيق للمستخدم ، ويوجد عدد قليل من البرامج تكون مهمة لكل إصدار من النواة .
هذا القسم يصف لنا عددا من الأدوات التي من المحتمل أن تكون مثبتة على نظام لينكس لديك.وإذا كنت تقوم بعمل ترقية للنواة إلى إصدار مختلفة عما هو مثبت في توزيعتك، فإن بعض هذه الحزم ربما تكون بحاجة إلى تحديث ليعمل النظام بشكل سليم.

util-linux

إن حزمة *util-linux* هي عبارة عن مجموعة صغيرة من الأدوات تقوم بعمل نطاق واسع من المهام المختلفة، وأغلب هذه الأدوات تعالج عملية ربط وإنشاء أقسام القرص الصلب ، وتتعامل أيضا مع توقيت النظام hardware clock .إذا كنت ترغب في تنزيل وتثبيت حزمة *util-linux* بنفسك يمكنك العثور عليه في .http://www.kernel.org/pub/linux/utils/util-linux
بداية من الإصدار 2.6.18 من النواة فإن إصدارة 2.10 من *util-linux* تعتبر أقدم نسخة تعمل بشكل سليم.ومن الموصى به أن تقوم بتثبيت آخر إصدار من هذه الحزمة، ذلك لأن النسخ الحديثة منها تدعم المميزات الجديدة المضافة إلى النواة.
ويعتبر Bind mounts أحد الأمثلة على الخيارات في الأنوية الحديثة، والنسخة الحديثة من *util-linux* لازمة لعملها على وجه صحيح.
وللتحقق من رقم إصدارة *util-linux* في نظامك اكتب الأمر التالي

```
$ fdformat –version
```

module-init-tools

تعتبر الحزمة module-init-tools لازمة إذا كنت ترغب في استخدام الوحدات البرمجية لنواة لينكس Linux kernel modules، ووحدة النواة kernel modul عبارة عن قطعة من الشيفرة قابلة للتحميل ويمكن إضافتها أو حذفها من النواة أثناء عمل النواة. ومن المفيد أن تقوم بعمل كومبايل لمشغلات الأجهزة device drivers على شكل وحدات modules ، ثم تقوم فقط بتحميل

ما يتناسب منها مع العتاد الموجود على النظام . كل توزيعات لينكس تستخدم modules لتحميل مشغلات العتاد والخيارات المطلوبة فقط بناء على العتاد الموجود على النظام ، بدلا من أن يكون مجبرا على بناء كل ما يمكنه من مشغلات وخيارات داخل النواة، في كتلة واحدة ضخمة.

ال modules الخاصة بالنواة توفر الذاكرة العشوائية عن طريق تحميل جزء الشفرة الذي تحتاجه فقط للتحكم بالجهاز بشكل سليم.

ولقد خضعت عملية تحميل وحدات النواة لتغيير جذري في إصدار النواة 2.6 ورابط الموديل (وهو الكود الذي يقوم بحل كل الرموز ويرسم كيفية وضع الأجزاء جنبا إلى جنب داخل الذاكرة العشوائية)، قد أصبح الآن مدمجا داخل النواة .وهو يجعل الأدوات الخاصة بفضاء المستخدم userspace tools أقل حجما .

تحتوي التوزيعات القديمة على حزمة تدعى *modutil والتي* لا تعمل بشكل سليم مع نواة 2.6 . وحزمة *module-init-tools* هي ما تحتاجه لجعل نواة 2.6 تعمل بشكل صحيح مع ال modules.

إذا كنت ترغب في تنزيل وتثبيت حزمة *module-init-tools* بنفسك يمكنك العثور عليها في

http://www.kernel.org/pub/linux/utils/kernel/module-init-tools

وبدءا من إصدار 2.6.18 من النواة، فإن الإصدار 0.9.10 من *module-init-tools* هو أقدم إصدار يمكنه العمل مع النواة بشكل سليم.ومن الموصى به أن تقوم بتثبيت آخر إصدار من هذه الحزمة،حيث إن المميزات الحديثة المضافة إلى النواة يمكنها العمل من خلال تلك الحزمة.إن عمل قائمة ممنوعات للموديلات غير المرغوب فيها لمنعها من التحميل تلقائيا باستخدام الحزمة *udev* يعتبر أحد الخيارات الموجودة في الإصدارات الحديثة ل module-init-tools، وليس القديم منها.وللتحقق من رقم إصدارة ,module-init-tools في نظامك اكتب الأمر التالي

```
$ depmod -V
```

أدوات تخصيص نظام الملفات

إن وجود نطاق واسع من أدوات تحديد نظم الملفات الخاصة لهو أمر ضروري لإنشاء صيغ وتهئية وإصلاح أقسام القرص الصلب. وحزمةutil-linux تحتوي بعضا من هذه الأدوات،ولكن بعضا من نظم الملفات المشهورة لديها حزم مستقلة تحتوي على البرامج الضرورية.

يعتبر نظام الملفات ext3 و نظام ext4 التجريبي ترقية لنظام ملفات ext2 ويمكن إدارتها بنفس الأدوات، ويمكن لأي نسخة قائمة على أساس ext2 أن تعمل مع النوعين الآخرين من نظم الملفات أيضا.

للعمل مع أي نوع من نطم الملفات هذه ، يجب أن يكون لديك الحزمة *e2fsprogs*. إذا كنت ترغب في تنزيل هذه الحزمة وتثبيتها بنفسك يمكنك الحصول عليها من http://e2fsprogs.sourceforge.net.

وبدءا من الإصدار 2.6.18 يعتبر الإصدار 1.29 من *e2fsprogs* هي أقدم نسخة يمكنها العمل مع النواة بشكل سليم .ومن الموصى به بشدة أن تقوم بتثبيت آخر إصدار من هذه الحزمة لتحصل عل مميزات متقدمة في نظم ملفات ext3 و ext4 وللتحقق من رقم إصدار *e2fsprogs* في نظامك اكتب الأمر التالي:

```
$ tune2fs
```

JFS

لاستخدام JFS المنتجة من قبل IBM يجب أن يكون لديك حزمة jfsutils .وإذا كنت ترغب في تحميل وتثبيت الحزمة بنفسك ، يمكنك الحصول عليها من http://jfs.sourceforge.net

للتحقق من رقم *إصدار jfsutil* على نظامك اكتب الأمر التالي :

```
$ fsck.jfs -V
```

ReiserFS

لاستخدام نظام ملفات ReiserFS يجب أن يكون لديك الحزمة *reiserfsprogs* . ، إذا كنت ترغب في تحميل وتثبيت الحزمة بنفسك ، يمكنك الحصول عليها من http://www.namesys.com/download.html

وبدءا من الإصدار 2.6.18 يعتبر الإصدار 3.6.3 من *reiserfsprogs* هي أقدم نسخة يمكنها العمل مع النواة بشكل سليم .

للتحقق من رقم *إصدار reiserfsprogs* على نظامك اكتب الأمر التالي :

```
$ reiserfsck -V
```

XFS

لاستخدام نظام ملفات XFS والمنتج من قبل SGI[1] يجب أن يكون لديك الحزمة xfsprogs*. إذا كنت ترغب في تحميل الحزمة وتثبيتها بنفسك، يمكنك تحميلها* من *http://oss.sgicom/projects/xfs*.

SGI ((1) اختصار لاسم الشركة Silicon Graphics, Inc هي شركة لتصنيع الحواسيب والحلول البرمجية عالية الكفاءة. ،تم تأسيس شركة سيليكون غرافيكس من قبل جيم كلارك في عام 1982 أساسا لتصنيع شاشات عرض الرسوميات الثلاثية الأبعاد. ومن أشهر منتجاتهم مكتبة الرسوميات المفتوحة open gl .

.إذا كنت ترغب في تحميل الحزمة وتثبيتها بنفسك، يمكنك تحميلها من *http://oss.sgicom/projects/xfs*.

وبدءا من الإصدار 2.6.18 يعتبر الإصدار 2.6.0 من *xfsprogs* هي أقدم نسخة يمكنها العمل مع النواة بشكل سليم .

للتحقق من ماهية إصدار *xfsprogs* المثبتة على نظامك اكتب ألأمر التالي:

```
$ xfs_db -V
```

Quotas

لاستخدام وظيفة الحصة quota في لينكس ، يجب أن يكون لديك الحزمة *quota-tools* (*) هذه الحزمة تشتمل على البرامج التي تتيح لك وضع حصص للمستخدمين ، وتزودك بإحصائيات عن الحصص المستخدمة من قبل مستخدمين مختلفين،وتصدر تحذيرات عندما يكون المستخدمون قاب قوسين أو أدنى من استهلاك الحصة الخاصة بهم من نظام الملفات.

إذا كنت ترغب في تحميل وتثبيت الحزمة بنفسك يمكن أن تجدها في http://sourceforge.net/projects/linuxquota

وبدءا من الإصدار 2.6.18 يعتبر الإصدار 3.09 من *quota-tools* هو أقدم نسخة يمكنها العمل مع النواة بشكل سليم .

للتحقق من ماهية إصدار *quota-tools* المثبتة على نظامك اكتب ألأمر التالي:

```
$ quota -V
```

NFS

لاستخدام نظام ملفات NFS بشكل صحيح ، فإنه يجب أن يكون لديك حزمة *nfs-utils* (**) هذ الحزمة تشتمل على برامج تتيح لك عمل ماونت لأقسام NFS كعميل client ، وتشغيل خادم NFS. إذا كنت ترغب في تحميل وتثبيت الحزمة بنفسك يمكن أن تجدها في *http://nfs.sf.net* .

للتحقق من ماهية إصدار *nfs-utils* المثبتة على نظامك اكتب الأمر التالي :

```
$ showmount –version
```

أدوات أخرى :

يوجد القليل من البرامج الأخرى المهمة لها علاقة وثيقة بنسخة النواة . هذه البرامج ليست دائما من متطلبات عمل النواة بشكل سليم ، ولكنها تقوم بتفعيل العديد من أنواع العتاد والوظائف.

(*)بعض التوزيعات لا سيما ديبيان، تسمى هذه الحزمة quota بدلا من *quota-tools*.

(**) بعض التوزيعات ، ول سيما ديبيان ، تسمى هذه الحزمة *nfs-common* بدل من *nfs-utils*.

udev هو برنامج يمكّن لينكس من تقديم تسمية للجهزة الموجودة على النظام في الدليل *dev/* وكذلك تزويد النظام بمجلد *dev/* الديناميكي، وهو يشبه إلى حد كبير نظام الملفات القديم *devfs* (تم حذفه الآن).

جميع توزيعات لينوكس تقريبا تستخدم *udev لإدارة* الدليل *dev/*، لذلك هو مطلوب لإقلاع النظام بشكل سليم. ولسوء الحظ ، فإن *udev يعتمد* على هيكلية ، *sys/* والتي كانت معروفة بأنها تتغير من وقت لآخر مع إصدارات النواة. وبعض هذه التغيرات التي حدثت في الماضي عرفت بأنها تحطم *udev* ، *حتى* إن جهازك لن يقلع على الوجه الصحيح. إذا كان لديك الإصدار الأخير من *udev المطلوب* لنواتك وحصلت على بعض المشكلات مع عملها بشكل صحيح، يرجى التصال بمطوري *udev على* قائمتهم البريدية والمتاحة في linux-hotplug-devel@lists.sourceforge.net.

من الموصى به بشدة أن تستخدم إصدارة *udev التي* جاءت مع توزيعة لينكس خاصتك . اذ أنها ترتبط داخل بتحديد عملية الإقلع بإحكام شديد، ولكن اذا كنت ترغب في تحديث *udev بنفسك* ، يمكنك ان تحصل عليها من : www.kernel.org/pub/linux/utils/kernel/hotplug/udev.html.

وبدءا من الإصدار 2.6.18 يعتبر الإصدار 081 من *udev* هو أقدم نسخة يمكنها العمل مع النواة بشكل سليم .ومن الموصى به أن تستخدم أحدث إصدار من *udev* حيث إنه يعمل بشكل أفضل مع الأنوية الجديدة ، نظرا إلى التغييرات في كيفية الاتصال بين udev وبين النواة.

للتحقق من ماهية إصدار *udev* المثبتة على نظامك اكتب ألأمر التالي:

```
$ udevinfo -V
```

Process tools

تشتمل الحزمة *procps الأدوات شائعة الاستخدام ps و top* وكذلك العديد من الأدوات البسيطة لإدارة ومراقبة العمليات العاملة على النظام.

إذا كنت ترغب في تحميل وتثبيت الحزمة بنفسك يمكنك الحصول عليها من http://procps.sourceforge.net .

وبدءا من الإصدار 2.6.18 من النواة ، يعتبر الإصدار 3.2.0 من *procps* هو أقدم نسخة يمكنها العمل مع النواة بشكل سليم .

للتحقق من ماهية إصدار *procps* المثبتة على نظامك اكتب ألأمر التالي:

```
$ ps –version
```

PCMCIA tools

من أجل عمل أجهزة PCMCIA[1] على النحو الصحيح مع نظام لينوكس، يجب استخدام برنامج مساعد لفضاء المستخدم userspace لإعداد الجهاز. بالنسبة للإصدارات الأقدم من نواة لينكس كان هذا البرنامج يدعى *pcmciautils* ، ولكن ذلك تم الاستعاضة عنه بنظام أبسط بكثير يدعى *pcmciautils*. إذا كنت ترغب في استخدام أجهزة PCMCIA ، يجب ان يكون لديك هذه الحزمة مثبتة لديك لتعمل هذه الأجهزة بشكل صحيح.

إذا كنت ترغب في تحميل وتثبيت الحزمة بنفسك يمكنك الحصول عليها من ftp://ftp.kernel.org/pub/linux/utils/kernel/pcmcia.

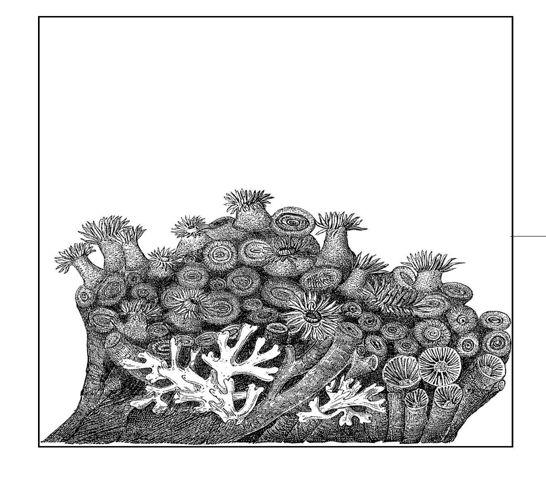
وبدءا من الإصدار 2.6.18 من النواة ، يعتبر الإصدار 004 من *pcmciautils* هو أقدم نسخة يمكنها العمل مع النواة بشكل سليم .ولكن ينصح بأحدث نسخة لإمكانية استخدام المميزات المتقدمة الجديدة في النظام الفرعي PCMCIA، مثل التحميل التلقائي لمشغل الأجهزة الجديدة الموجودة.

للتحقق من ماهية إصدار *pcmciautils* المثبتة على نظامك اكتب ألأمر التالي:

```
$ pccardtl -V
```

---

(1) Pcmcia : هي اختصار ل Personal Computer Memory Cardp International Association وهي مؤسسة أنشأت سنة 1989 من أجل توحيد أنماط اتصال الجهزة وبطاقات التوسعة مع الكمبيوتر المحمول ومن أمثلتها بطاقات الذاكرة و البطاقات التي يمكن استخدامها لغراض التصال اللسلكي ، والمودم وغيرها من الوظائف في اجهزة الكمبيوتر المحمول -للمزيد من التفاصيل حول هذه المؤسسة ومنتجاتها يمكنك الاطلاع على الموقع الرسمي http://pcmcia.org/about.htm



3

الحصول على الملف المصدري للنواة

عندما ترغب في بناء النواة الخاصة بك فأنت تريد آخر إصدارة مستقرة منها .وهناك العديد من التوزيعات تأتي بحزمها الخاصة من الملف المصدري للنواة، ولكن هذا أمر نادر، فأغلب التوزيعات الحديثة لا تفعل ذلك في الإصدارات الحالية. فحزم التوزيعات تحتوي على ميزة وهي أنها أصبحت مبنية بما يتلاءم مع المترجم الخاص بها والأدوات الأخرى التي تأتي مع التوزيعة (وقد شرح الفصل الثاني أهمية هذه التوافقية)ولكنها قد لا تقدم لك الحد الأقصى من الأداء أو الوظائف التي تريدها.

إذا كان يمكنك خلق البيئة الخاصة بك مع آخر نواة ، ومترجم ، وغيرها من الادوات ، فستكون قادرا على بناء ما تريد بالضبط . ويركز هذا الفصل على تحديد أي ملف مصدري للنواة عليك تحميله، وكيفية الحصول عليه.

شجرة النواة التي عليك استخدامها

قديما كانت نواة لينكس تتفرع إلى فرعين فحسب:

1- فرع "التطوير"- development branch .

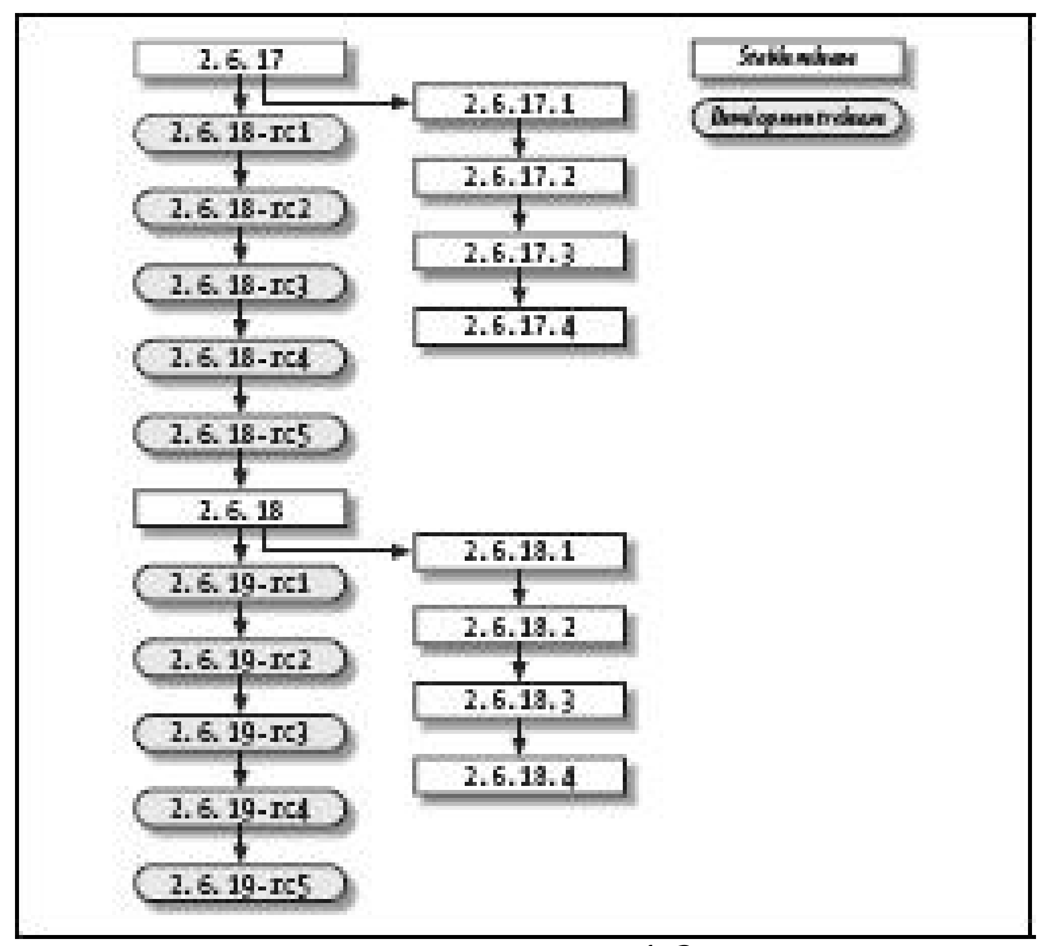
2- فرع النواة "المستقرة" stable branch.

فرع التطوير *development branch* كان يستدل عليه عن طريق الرقم الفردي في الرقم الثاني من الإصدارة . بينما الفرع المستقر *stable branch* من النواة يستخدم الأرقام الزوجية،

لذلك - على سبيل المثال - تعتبر الإصدارة 2.5.25 نواة تحت التطوير، بينما إصدارة 2.4.25 إصدارة مستقرة.

ولكن بعد إنشاء سلسلة 2.6 من النواة قرر مطورو النواة تجاهل هذه الطريقة من التقسيم على شجرتين منفصلتين ، وأعلنوا أن كل إصدارات سلسلة النواة من طراز

2.6 تعتبر stable ،ولا يهم سرعة وتيرة التطوير الذي يحدث للنواة.
ولقد أتاحت الشهور القليلة فيما بين إصدارات الرقم الرئيس 2.6 للمطورين أن يضيفوا المزايا الجديدة، ومن ثم ترسيخها وتثبيتها في الإصدارة التالية.
بالإضافة الى ذلك، فقد تم إنشاء فرع النواة "stable -" بحيث تطلق إصدارات تحتوي على ترقيع وإصلاح للأخطاء البرمجية bugs وتحديثات أمنية للنواة القديمة.
وهذا وأفضل شرح مع بعض الأمثلة، كما يتبين في الشكل 3-1.
قام فريق تطويرالنواة بإطلاق إصدار النواة 2.6.17 بوصفها إصدار مستقر. ومن ثم بدأ المطورون العمل على ميزات جديدة، وبدءوا في إصدارات rc- على إنها إصدارات قيد التطوير، ولذا فإن الناس يستطيعون المساعدة من خلال اختبار النواة وتصحيح الأخطاء البرمجية لهذه التغيرات .
وبعد موافقة كل شخص على تلك النسخة المطورة أصبحت بذلك مستقرة بشكل كاف، وتم إطلاق إصدارة الكيرنل رقم 2.6.18 .
وهذه هي الدورة التي تأخذها النواة عادة خلال شهرين أو ثلاثة ويعتمد ذلك على مجموعة متنوعة من العوامل.



شكل 3-1: دورة تطوير إصدارة النواة

وبينما يجري تطوير سمات الجديدة للنواة؛ تم إطلاق الإصدارين 2.6.17.1 ، 2.6.17.2 ، وغيرها من الإصدارات المستقرة للنواة، متضمنة على إصلاحات وتحديثات أمنية.

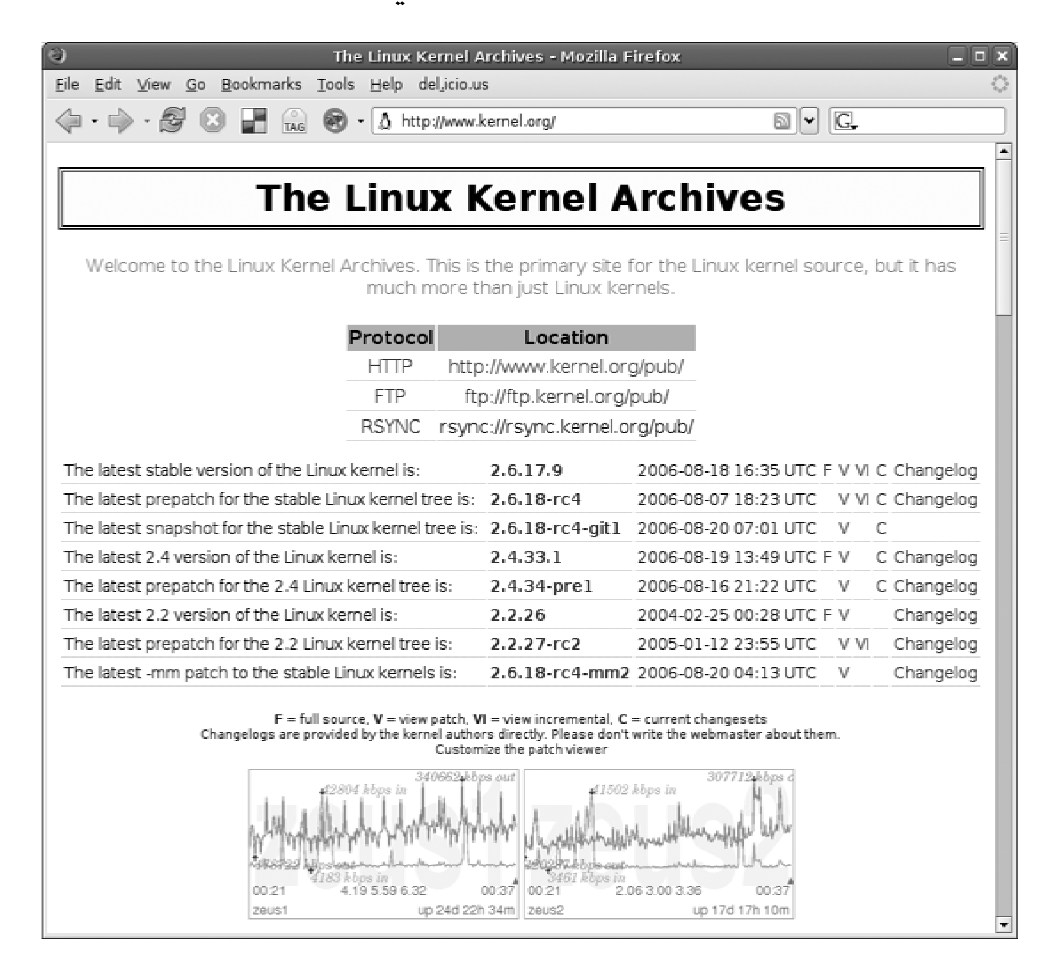
اذا كنت ترغب فقط في استخدام أحدث نواة لعملك ، فمن الموصى به استخدام إصدارة نواة مستقرة.

اذا كنت ترغب في مساعدة مطوري نواة في اختبار مميزات الإصدار المقبل للنواة واعطائهم مقترحاتك، استخدم إصدارة النواة تحت التطوير. ونحن نفترض أنك تستخدم إصدارة نواة مستقرة.

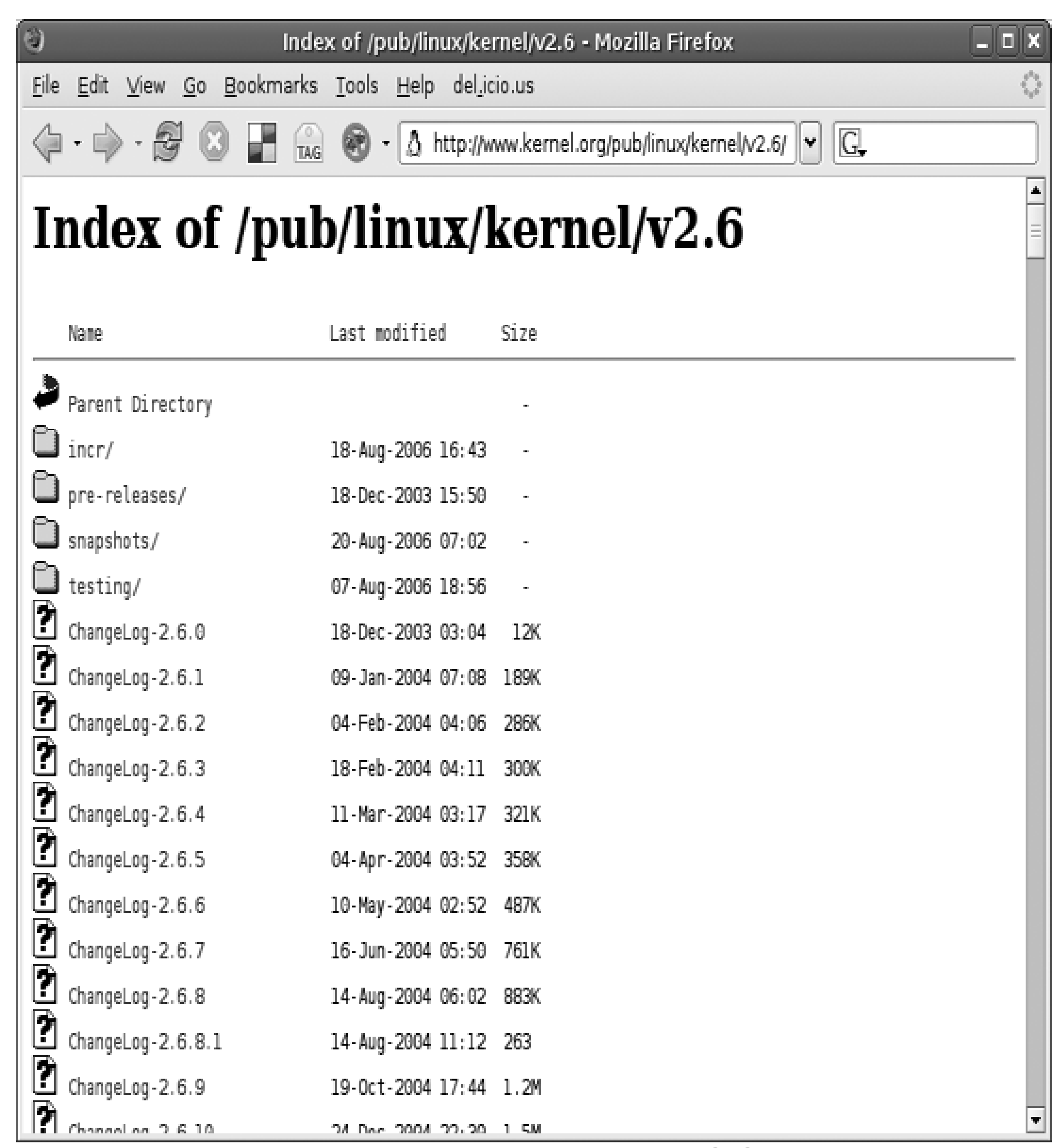
أين يمكن الحصول عل الملف المصدري للنواة

جميع الملفات المصدرية للنواة يمكن أن توجد في أحد مواقع *kernel.org* على شبكة الإنترنت، والخوادم الخاصة بالملفات المصدرية للنواة، حيث تتيح لكل شخص بأن يعثر على أقرب خادم لمنطقته الجغرافية والتحميل منه. وذلك يسمح للخوادم الرئيسة للكيرنل سريع الاستجابة مع مواقع المرآة mirror sites، ويتيح المستخدمين تحميل ما يحتاجون من الملفات بأسرع قدر ممكن.

الموقع الرئيس *http://www.kernel.org* يعرض كل إصدارات الكيرنل لسلاسل متنوعةمن الأنوية المختلفة، كما يظهر في الشكل 3-2



لتنزيل أحدث نسخة مستقرة من النواة اضغط على الحرف F في السطر الخاص بإصدارة النواة. وذلك سوف يحمل سلسلة الملف المصدري للنواة كاملا. أو يمكنك الإبحار في الدليل الفرعي المناسب لكل إصدارات سلسلة نواة 2.6 في هذه الصفحة: http://www.us.kernel.org/pub/linux/kernel/v2.6 كما هو مبين في الشكل في الشكل 3-3



Name	Last modified	Size
Parent Directory		-
incr/	18-Aug-2006 16:43	-
pre-releases/	18-Dec-2003 15:50	-
snapshots/	20-Aug-2006 07:02	-
testing/	07-Aug-2006 18:56	-
ChangeLog-2.6.0	18-Dec-2003 03:04	12K
ChangeLog-2.6.1	09-Jan-2004 07:08	189K
ChangeLog-2.6.2	04-Feb-2004 04:06	286K
ChangeLog-2.6.3	18-Feb-2004 04:11	300K
ChangeLog-2.6.4	11-Mar-2004 03:17	321K
ChangeLog-2.6.5	04-Apr-2004 03:52	358K
ChangeLog-2.6.6	10-May-2004 02:52	487K
ChangeLog-2.6.7	16-Jun-2004 05:50	761K
ChangeLog-2.6.8	14-Aug-2004 06:02	883K
ChangeLog-2.6.8.1	14-Aug-2004 11:12	263
ChangeLog-2.6.9	19-Oct-2004 17:44	1.2M

شكل 3-3 : الحصول على الملف المصدري للنواة

ومن الممكن أيضا تحميل مصدر النواة باستخدام سطر الأوامر ، باستخدام الأمر *wget* او *curl* ، وكلا الأمرين ينبغي أن ياتيا مع توزيعة لينكس الخاصة بك. لتحميل النسخة 2.6.17.8 من النواة باستخدام wget ، اكتب :

```
$ wget http://www.kernel.org/pub/linux/kernel/v2.6/
linux-2.6.17.8.tar.gz
    --17:44:55--
    http://www.kernel.org/pub/linux/kernel/v2.6/li
    nux-2.6.17.8. tar.gz
            => `linux-2.6.17.8.tar.gz'
    Resolving www.kernel.org... 204.152.191.5,
    204.152.191.37
    Connecting to www.kernel.org|
    204.152.191.5|:80... connected.
    HTTP request sent, awaiting response... 200 OK
    Length: 51,707,742 (49M) [application/x-gzip]
    100%[=================================================>]
    51,707,742 35.25K/s
    ETA 00:00
18:02:48 (47.12 KB/s) - `linux-2.6.17.8.tar.gz' saved
[51707742/51707742]
```

للتنزيل باستخدام الأمر *curl* :

```
$ curl
http://www.kernel.org/pub/linux/kernel/v2.6/linux-2.6.17.8.tar.gz \
-o linux-2.6.17.8.tar.gz
  % Total    % Received % Xferd Average Speed    Time
Time     Time
Current
                                 Dload Upload    Total
Spent    Left
Speed
100 49.3M 100 49.3M     0     0 50298       0 0:17:08
0:17:08 --:--:--
100k
```

للحصول على أسهل وأسرع طريقة لتحديد آخر إصدارة من النواة، استخدم المعلومات المتاحة على الموقع الإلكتروني:

http://www.kernel.org/kdist/finger_banner

كما هو موضح في الشكل 4-3

شكل 4-3:معرفة آخر إصدار من النواة

Mozilla Firefox

File Edit View Go Bookmarks Tools Help del.icio.us

http://www.kernel.org/kdist/finger_banner

```
The latest stable version of the Linux kernel is:              2.6.17.9
The latest prepatch for the stable Linux kernel tree is:       2.6.18-rc4
The latest snapshot for the stable Linux kernel tree is:       2.6.18-rc4-git1
The latest 2.4 version of the Linux kernel is:                 2.4.33.1
The latest prepatch for the 2.4 Linux kernel tree is:          2.4.34-pre1
The latest 2.2 version of the Linux kernel is:                 2.2.26
The latest prepatch for the 2.2 Linux kernel tree is:          2.2.27-rc2
The latest -mm patch to the stable Linux kernels is:           2.6.18-rc4-mm2
```

ماذا نصنع بالملف المصدري :

الآن وقد قمت بتنزيل الملف المصدري المناسب للنواة، أين تقترح أن نذهب ؟

نحن نقترح أن تنشئ مجلدا في الدليل المحلي الخاص بك "home" يسمى linux لاحتواء مختلف الملفات المصدرية للنواة :

$ mkdir ~/linux

والآن قم ينقل الملف المصدري إلى ذلك المجلد الذي أنشأته سالفا

$ mv ~/linux-2.6.17.8.tar.gz ~/linux/

ثم اذهب إلى داخل المجلد المذكور :

$ cd ~/linux

$ ls

```
linux-2.6.17.8.tar.gz
```

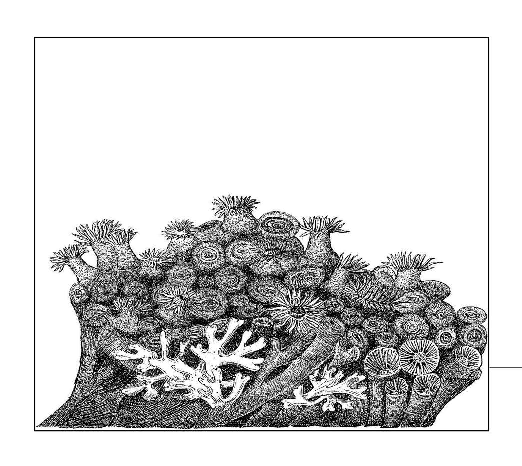
والآن أصبح الملف المصدري داخل الدليل الصحيح، قم بفك أرشفة شجرة الملف المصدري :

$ tar -xzvf linux-2.6.17.8.tar.gz

سوف تمتلئ الشاشة بالملفات التي تم فك أرشفتها، وأنت ما زلت داخل الدليل ‎/linux **اكتب:**

$ ls

```
linux-2.6.17.8.tar.gz
linux-2.6.17.8/
```





الإعــداد والبنــاء

الآن وقد قمت بتحميل الملف المصدري لإصدار النواة الذي اخترته ووضعته داخل أحد الأدلة المحلية، حان الآن وقت بناء الشفرة.
أول خطوة هي إعداد النواة مع الخيارات المناسبة، وبعدها يمكن عمل ترجمة compile للنواة. كلا المهمتين يمكن فعلهما من خلال الأداة القياسية *make* .

إنشاء ملف الإعداد :

عملية تهيئة -أو إعداد- النواة محفوظة في ملف يسمى *config*. في أعلى مجلد في شجرة الملف المصدري للنواة .
إذا كنت تقوم بفك ملف مصدر النواة فحسب فلن يكون هناك ملف config.
ولذلك يلزمك إنشاؤه.ويمكن إنشاء هذا الملف من الصفر عن طريق تأسيسه على الإعدادات الافتراضية المستمدة من إصدار النواة التي تعمل الآن، أو مأخوذا من إصدارة لأحد توزيعات لينكس. وسوف نقوم بتغطية أول طريقتين هنا، وأما الطريقتان الأخيرتان ففي الفصل السابع.

: Configuring from Scratch

أبسط الطرق المستخدمة في تهيئة النواة هي طريقة *make config* :

```
$ cd linux-2.6.17.10
$ make config
  make config
scripts/kconfig/conf arch/i386/Kconfig
*

* Linux Kernel Configuration
```

```
*
*
* Code maturity level options
*
Prompt for development and/or incomplete code/drivers
(EXPERIMENTAL) [Y/n/?]
Y
*
* General setup
*
Local version - append to kernel release (LOCALVERSION)
[]
Automatically append version information to the version
string
(LOCALVERSION_AUTO) [Y/n/?] Y
...
```

برنامج إعداد النواة سوف يتجول خلال كل خيارات الإعداد، ويسألك إذا كنت تريد تمكين هذا الخيار أم لا، وبالضبط، سوف تكون اختياراتك لكل خيار معروضة بصيغة [?/Y/m/n]. الحرف الكبير Y هو الخيار الافتراضي ويمكنك اختياره عن طريق الضغط على مفتاح Enter فحسب. والخيارات الأربعة هي كما يلي :

- *Y* : البناء مباشرة داخل النواة .
- n : جعله خارج النواة.
- m : بناؤه كأحد الموديلات لتحميله عند الحاجة إليه.
- ? : عرض رسائل مساعدة موجزة عن الخيار والعودة للمحث مرة أخرى.

تحتوي النواة تقريبا على ألفين من خيارات الإعداد المختلفة، لذلك فإن السؤال عن كل خيار على حدة يأخذ الكثير جدا من الوقت. ولحسن الحظ فإن هناك طريقة أخرى لإعداد النواة : التهيئة بناءًا على تهيئة سابقة.

Default Configuration Options

كل إصدار من النواة يأتي معه إعدادات افتراضية للنواة، هذه الإعدادات مبنية بشكل فضفاض على الوضع الافتراضي الذي يشعر المسؤول عن النواة والقائم على بناء هذه الهرمية بأنه يقدم أفضل خيارات يمكن استخدامها. وفي بعض الحالات ما هي إلا الإعدادات التي يستخدمها القائم على النواة بنفسه على جهازه الشخصي .وذلك واقع مع معمارية المعالجات i386 حيث إن الإعدادات الافتراضية للنواة يتطابق تقريبا مع ما كان يستخدمه لينوس تورفالدز على جهاز التطوير الخاص به.

لإنشاء هذه الإعدادات الافتراضية قم بعمل الآتي:

$ cd linux-2.6.17.10

$ make defconfig

يوجد عدد ضخم من خيارات الإعداد سوف تنسدل بسرعة على الشاشة وسوف يتم كتابة الملف *config*. ويوضع في مجلد النواة.

الآن قد تم تهيئة النواة بنجاح، ولكن يجب تعديل خواصها بما يناسب جهازك للتأكد من عملها بشكل صحيح.

تعديل ملف الإعداد :

الآن لدينا الملف الأساسي للإعداد الذي تم إنشاؤه، ويجب تعديله ليدعم العتاد الموجود على النظام. لمزيد من التفاصيل عن خيارات الإعداد التي تحتاج لإنجازها، يرجى مطالعة الفصل السابع.

هنا سوف نوضح لك كيفية انتقاء هذه الخيارات التي ترغب في تغييرها.

هناك ثلاثة أنواع تفاعلية مختلفة من أدوات إعداد النواة:

- طريقة من خلال الترمنال تسمى *menuconfig*.
- طريقة رسومية مبنية على GTK+ تسمى *gconfig* .
- طريقة رسومية مبنية على QT تسمى *xgonfig*.

: Console Configuration Method

طريقة *menuconfig* لإعداد الكيرنل هي برنامج كونسول، تقدم طريقة للتجول حول إعدادات النواة باستخدام مفاتيح الأسهم من لوحة المفاتيح. لبدء هذه الطريقة من الإعداد ، اكتب:

$ make menuconfig

سوف ترى شاشة تشبه كثيرا الشكل 1-4

```
Linux Kernel Configuration
Arrow keys navigate the menu.  <Enter> selects submenus --->.
Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
<M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </>
for Search.  Legend: [*] built-in  [ ] excluded  <M> module  < >

    Code maturity level options  --->
    General setup  --->
    Loadable module support  --->
    Block layer  --->
    Processor type and features  --->
    Power management options (ACPI, APM)  --->
    Bus options (PCI, PCMCIA, EISA, MCA, ISA)  --->
    Executable file formats  --->
    Networking  --->
    Device Drivers  --->
v(+)

        <Select>    < Exit >    < Help >
```

التعليمات الخاصة بالإبحار خلال البرنامج ، ومعاني الرموز المختلفة، تظهر في أعلى الشاشة. بقية الشاشة تتضمن مختلف خيارات تهيئة النواة.

برنامج إعداد النواة ينقسم إلى أقسام، وكل قسم يتضمن الخيارات التي تتوافق مع موضوع محدد. داخل تلك الفروع يمكن وجود أقسام فرعية مخصصة لمختلف المواضيع. وكمثال على ذلك ، جميع مشغلات الأجهزة يمكن العثور عليها تحت القائمة الرئيسية لخيار Device Drivers. للدخول إلى هذه القائمة، حرك مفتاح السهم الأسفل تسع مرات حتى يضيء سطر Device Drivers ---->
كما هو مبين في الشكل 4-2.

```
Linux Kernel Configuration
Arrow keys navigate the menu.  <Enter> selects submenus --->.
Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
<M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </>
for Search.  Legend: [*] built-in  [ ] excluded  <M> module  < >
^(-)
    General setup  --->
    Loadable module support  --->
    Block layer  --->
    Processor type and features  --->
    Power management options (ACPI, APM)  --->
    Bus options (PCI, PCMCIA, EISA, MCA, ISA)  --->
    Executable file formats  --->
    Networking  --->
    Device Drivers  --->
    File systems  --->
v(+)

        <Select>    < Exit >    < Help >
```

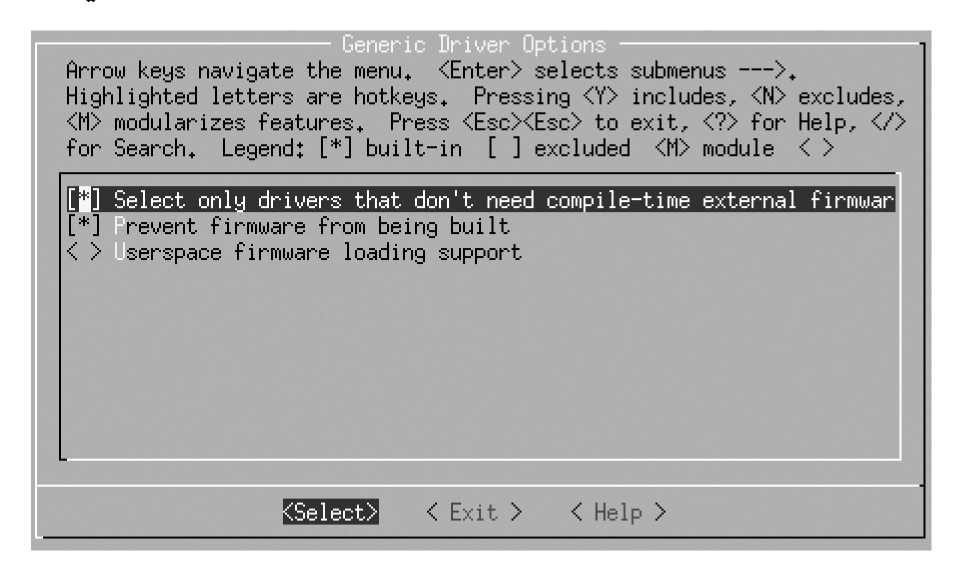
بعد ذلك اضغط مفتاح Enter وسوف ينتقل داخل القائمة الفرعية ل Device Drivers ويعرضها كما يتضح ذلك في الشكل 4-3

```
Device Drivers
Arrow keys navigate the menu.  <Enter> selects submenus --->.
Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
<M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </>
for Search.  Legend: [*] built-in  [ ] excluded  <M> module  < >

    Generic Driver Options  --->
    Connector - unified userspace <-> kernelspace linker  --->
    Memory Technology Devices (MTD)  --->
    Parallel port support  --->
    Plug and Play support  --->
    Block devices  --->
    ATA/ATAPI/MFM/RLL support  --->
    SCSI device support  --->
    Multi-device support (RAID and LVM)  --->
    Fusion MPT device support  --->
v(+)

        <Select>    < Exit >    < Help >
```

يمكنك الاستمرار في التحرك لأسفل خلال القائمة الهرمية بنفس الطريقة . ولكي ترى خيارات القائمة الفرعية Generic Driver Options اضغط مفتاح Enter مرة أخرى وسوف ترى الثلاث خيارات كما هو مبين في الشكل 4-4



شكل 4-4 : القائمة الفرعية Generic Driver Options

الخيار الأول والثاني أمامه علامة [*] . وهذا يعني أن هذا الخيار هو المختار (بحكم أن العلامة * توجد في منتصف الأقواس المربعة [ ]) ، وأن هذا الخيار هو خيار نعم أو لا. الخيار الثالث أمامه علامة < > وهي تبين أن هذا الخيار يمكن أن يبنى داخل النواة (y) ، أو يبني على شكل موديل (M) ، أو يستبعد بالكلية خارج النواة (N).
اذا كان الخيار المختار مع ضغط الحرف y، فسوف تحتوي الأقواس <> على العلامة *. وإذا كان الخيار هو موديل مع الحرف M فسوف تحتوي الأقواس على

الحرف M ، وإذا كان الخيار هو التعطيل مع ضغط الحرف N سوف يعرض فقط أقواس فارغة.

لذا ، إذا كنت ترغب في تغيير هذه الخيارات الثلاثة لاختيار المشغلات التي لا تحتاج إلى برنامج ثابت -firmware- خارجي أثناء عملية الكومبايل، قم بتعطيل الخيار لمنع ال firmware من البناء، وبناء محمل ال userspace firmware كـ module، اضغط Y للخيار الأول، و N للخيار الثاني، و M للخيار الثالث.

واجعل الشاشة كما يبدو في الشكل 4-5.

```
                    Generic Driver Options
 Arrow keys navigate the menu.  <Enter> selects submenus --->.
 Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
 <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </>
 for Search.  Legend: [*] built-in  [ ] excluded  <M> module  < >

 [*] Select only drivers that don't need compile-time external firmwar
 [ ] Prevent firmware from being built
 <M> Userspace firmware loading support

              <Select>    < Exit >    < Help >
```

شكل 4-5 : تغيير القائمة الفرعية ل Generic Driver Options

بعد انتهائك من التغييرات التي تريدها على هذه الشاشة، اضغط إما مفتاح Escape أو مفتاح السهم الأيمن متبوعا بمفتاح Enter لمغادرة هذه القائمة الفرعية .

كل هذه الخيارات المختلفة للنواة يمكن استكشافها على هذا النحو.

عند انتهائك من عمل التغييرات التي تريد عملها لإعداد النواة ، اخرج من البرنامج عن طريق الضغط على مفتاح Escape من القائمة الرئيسية.

سوف يعرض أمامك الشاشة الموجودة في الشكل 4-6 التي تسألك عما إذا كنت تريد حفظ تغييراتك لملف إعداد النواة.

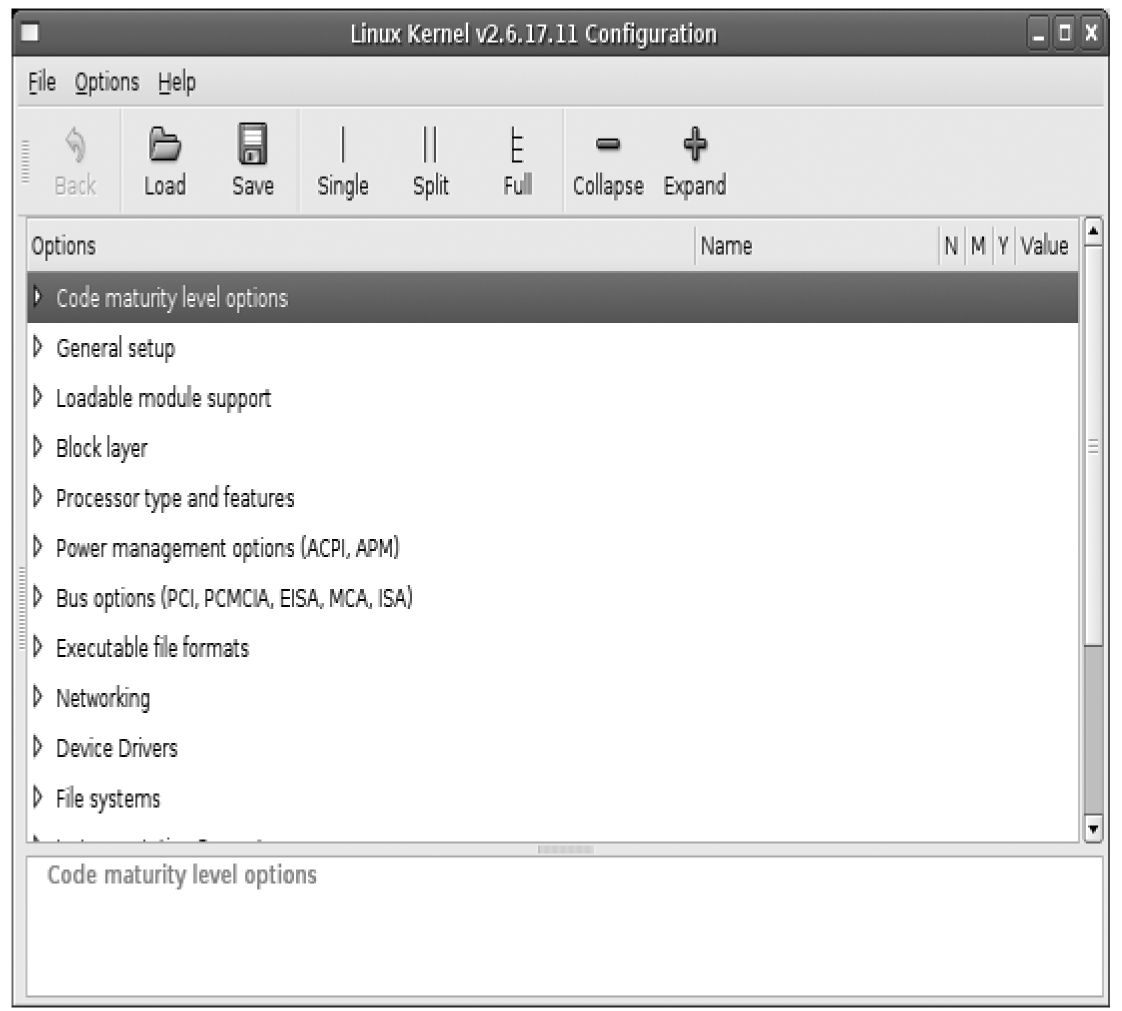


الشكل 4-6 : حفظ خيارات إعداد النواة

اضغط مفتاح Enter لحفظ الإعدادات ، أو إن كنت ترغب في إلغاء أي تعديلات قد أجريت،فاضغط السهم الايمن للانتقال إلى الخيار <No> وبعد ذلك اضغط مفتاح Enter.

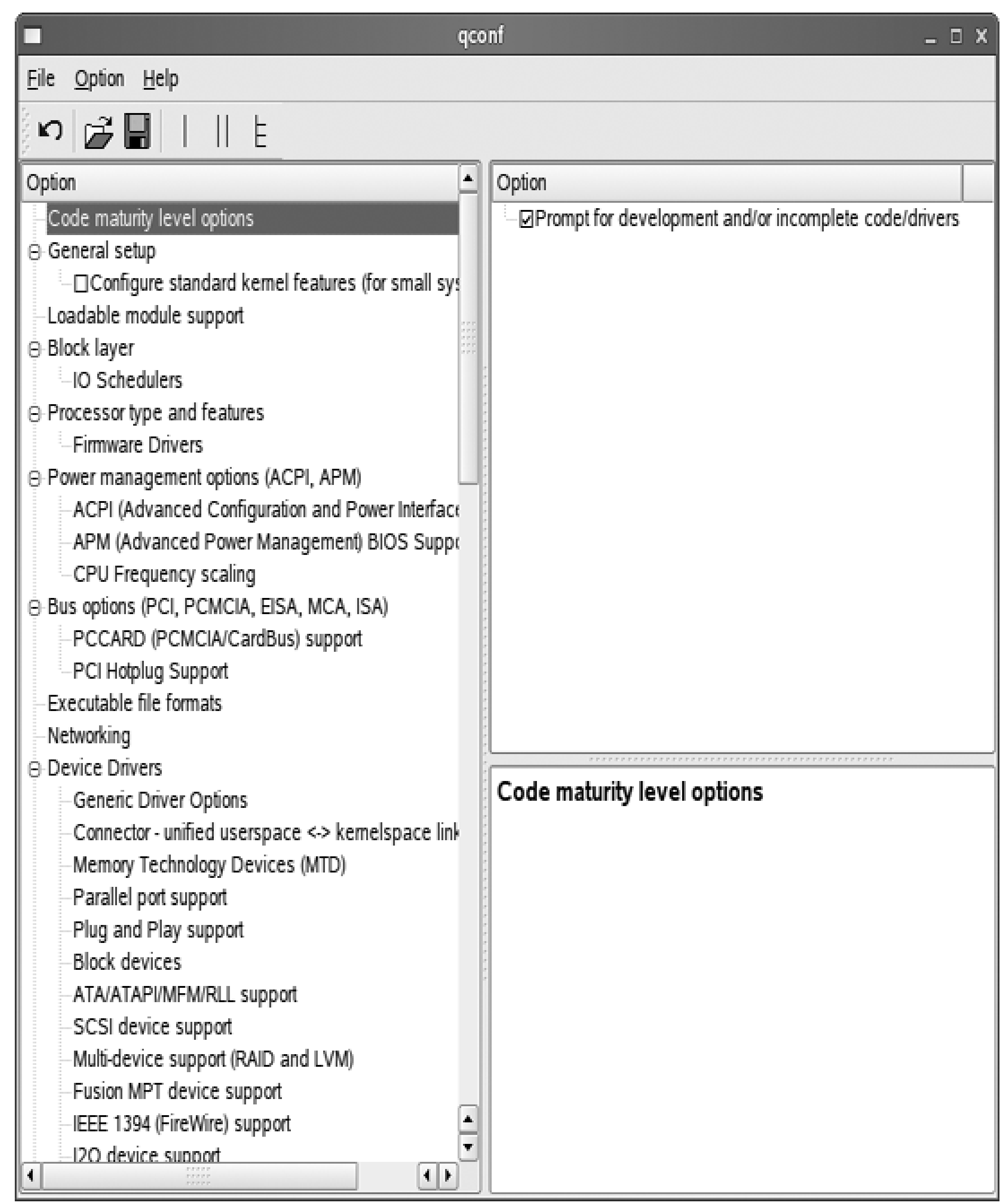
الطرق الرسومية للإعداد Graphical Configuration Methods

تستخدم الطريقتان *gconfig* و *xconfig* برنامجا رسوميا يتيح لك التعديل على إعدادات النواة. هاتان الطريقتان متطابقتان تقريبا ، والفرق الوحيد هو الأدوات التي كتبت بها. حيث إن *gconfig* مكتوب باستخدام +GTK والشاشة مكونة من لوحتين كما يبدو في الشكل 4-7



شكل 4-7 : عمل الشاشة *gconfig*

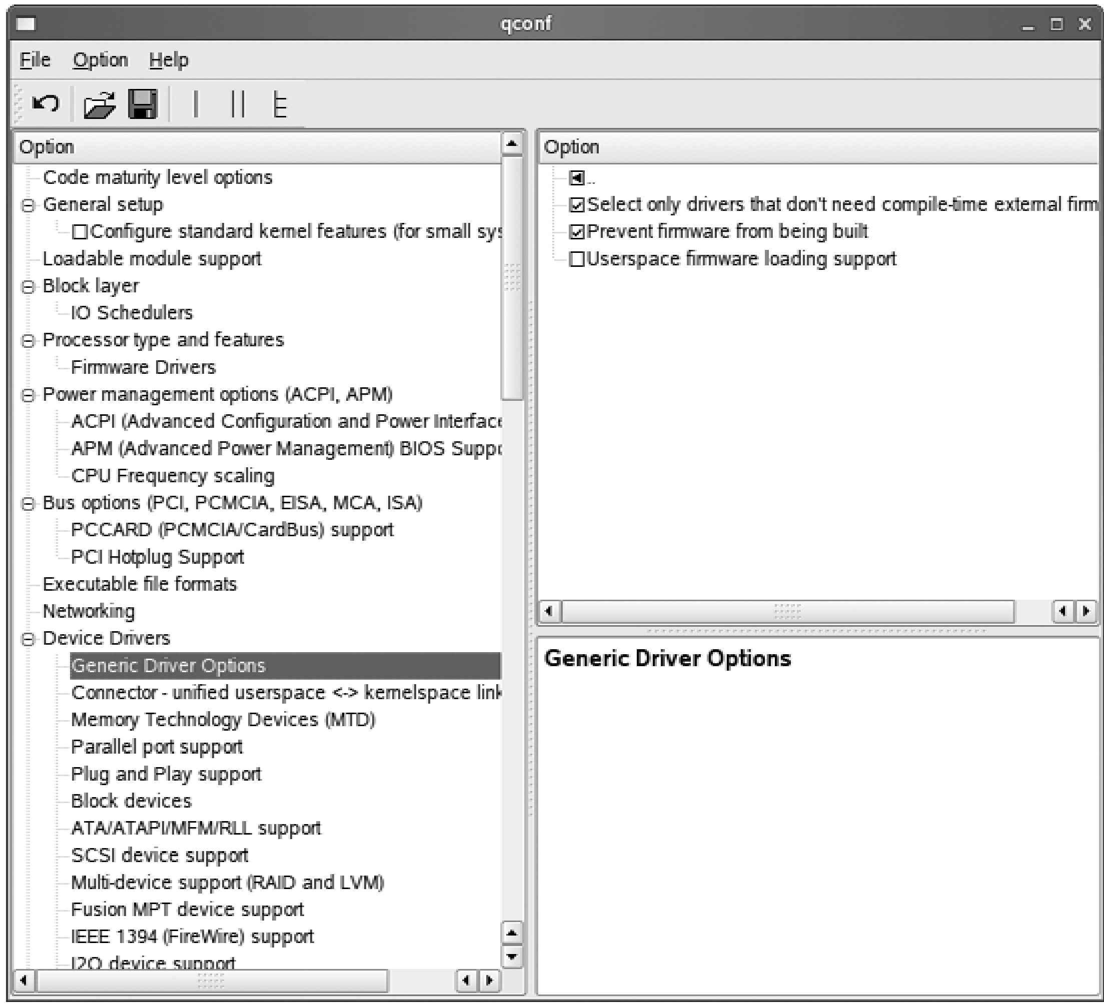
وطريقة *xconfig* مكتوبة بالأداة *QT* وتحتوي على ثلاث لوحات في الشاشة كما يتضح في الشكل 4-8 .



الشكل 8-4 : عمل الشاشة *xconfig*

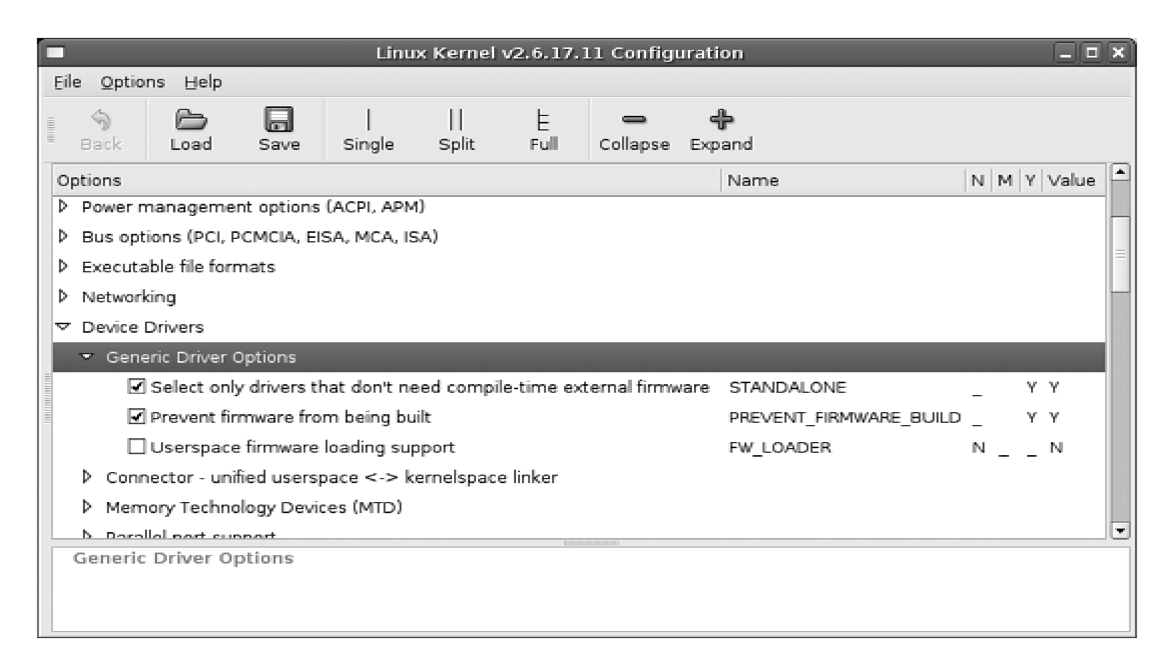
استخدم الفارة للتجول في القوائم الفرعية وتحديد الخيارات.

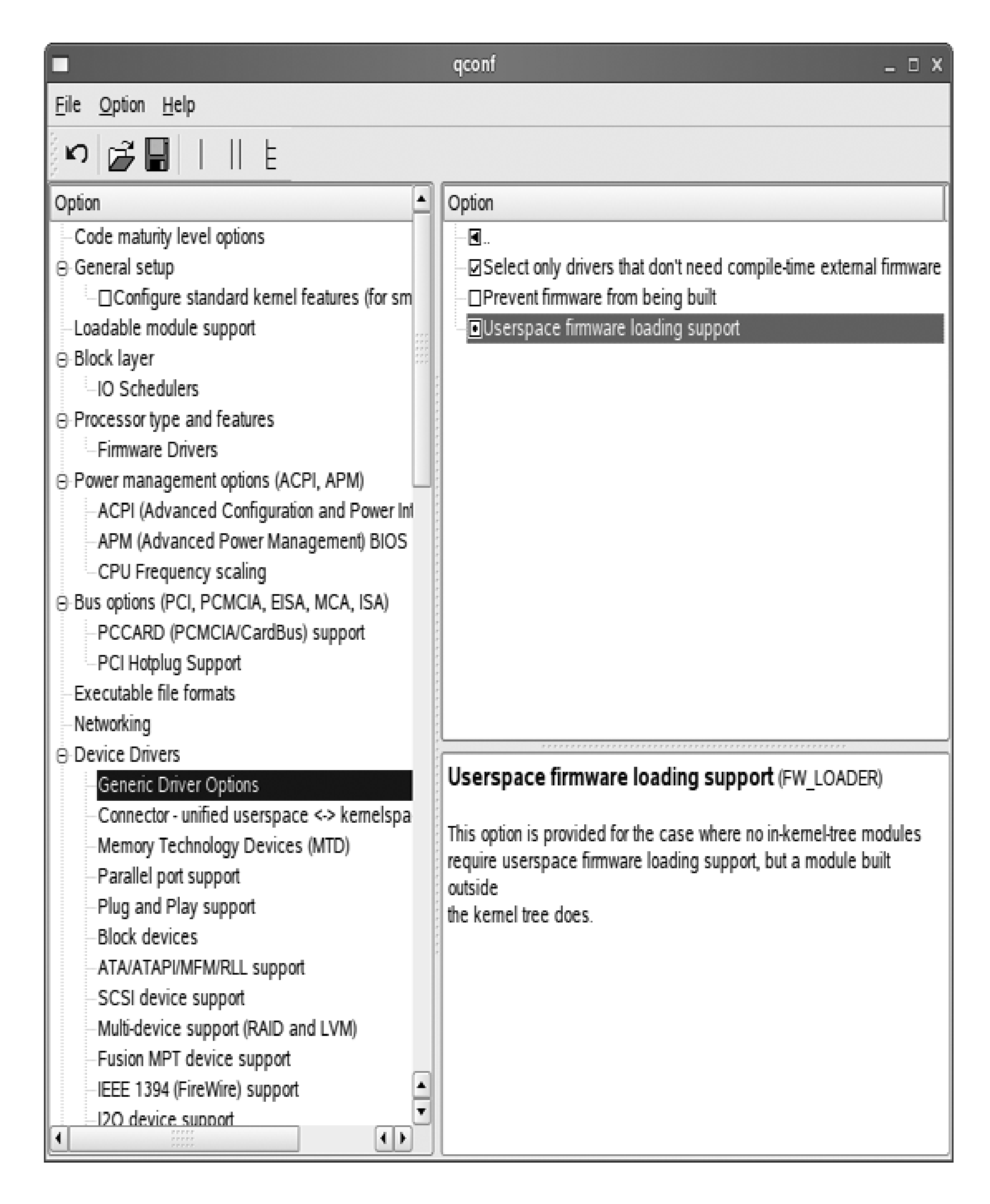
على سبيل المثال، يمكنك استخدامها كما في الشكل 8-4 لاختيار القائمة الفرعية Generic Driver Options للقائمة الرئيسية Device Drivers وسوف تتغير لتبدو مثل الشكل 9-4.



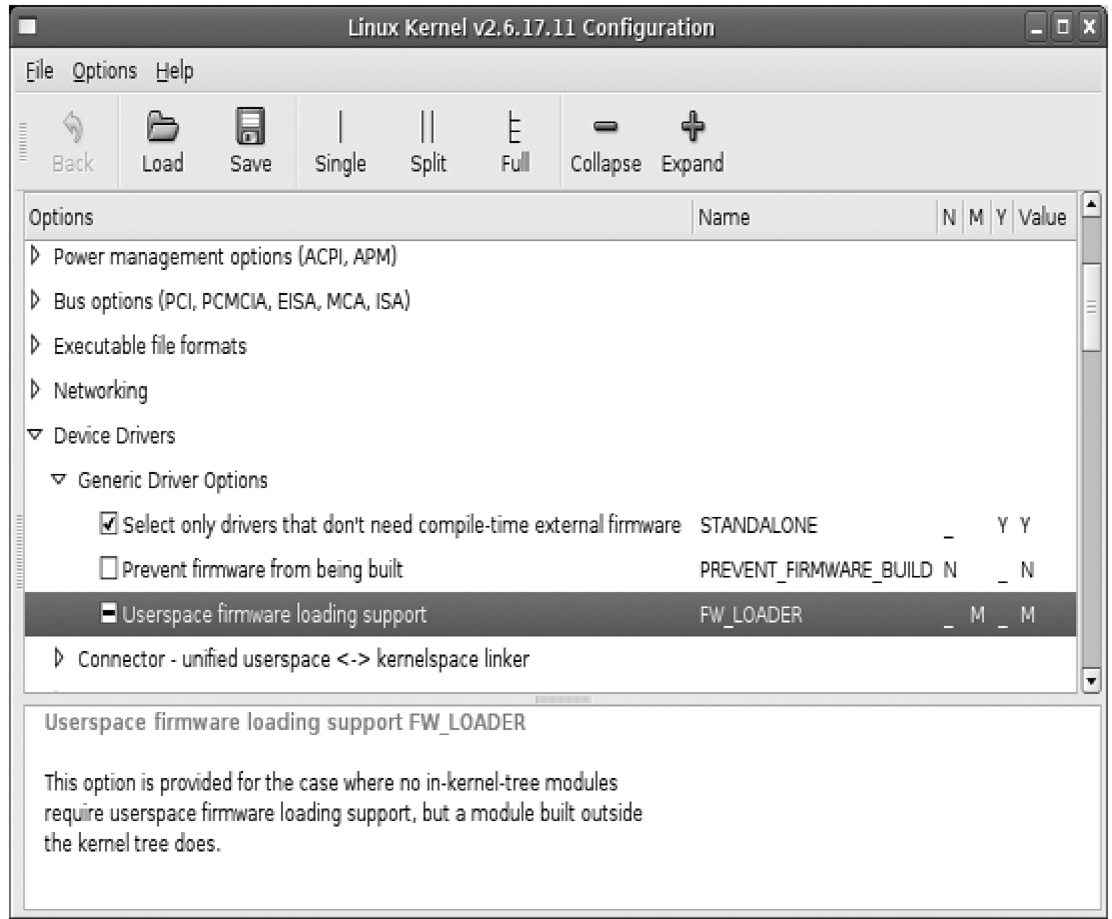
شكل 9-4 : عمل xconfig ل Generic Driver Options

وفي المقابل فإن شاشة gconfig تبدو كما في الشكل 10-4





شكل 11-4: make xconfig Generic Driver Options changed



شكل 4-12: make gconfig Generic Driver Options changed

بناء النواة:

الآن وقد قمت بعمل تهيئة للنواة التي ترغب في استخدامها، أنت الآن تحتاج لبناء النواة. وذلك بعبارة بسيطة يعني كتابة أمر واحد :

```
$ make
CHK      include/linux/version.h
 UPD      include/linux/version.h
 SYMLINK include/asm -> include/asm-i386
 SPLIT    include/linux/autoconf.h -> include/config/*
 CC        arch/i386/kernel/asm-offsets.s
 GEN      include/asm-i386/asm-offsets.h
 CC        scripts/mod/empty.o
 HOSTCC  scripts/mod/mk_elfconfig
 MKELF    scripts/mod/elfconfig.h
 HOSTCC  scripts/mod/file2alias.o
 HOSTCC  scripts/mod/modpost.o
 HOSTCC  scripts/mod/sumversion.o
```

```
HOSTLD  scripts/mod/modpost
HOSTCC  scripts/kallsyms
HOSTCC  scripts/conmakehash
HOSTCC  scripts/bin2c
CC            init/main.o
CHK include/linux/compile.h
UPD include/linux/compile.h
CC  init/version.o
CC  init/do_mounts.o
   ...
```

تشغيل برنامج *make* يتسبب في استخدام نظام بناء النواة للإعدادات التي قمت باختيارها لبناء النواة، وجميع الوحدات البرمجية modules اللازمة لدعم هذه الإعدادات(*). وبينما تتم عملية بناء النواة ، يقوم برنامج *make* بعرض أسماء الملفات واحدا تلو الآخر، وما يحدث له حاليا ، إلى جانب أي تحذيرات أو أخطاء متعلقة بالبناء. فإذا تم الانتهاء من بناء نواة دون أي أخطاء، تكون نجحت في إنشاء صورة النواة. ومع ذلك، هناك حاجة إلى تركيبها على النحو الصحيح قبل محاولة الإقلاع . انظر الفصل 5 لكيفية القيام بذلك.

ومن الأمور العادية جدا أن تحصل على أخطاء عند بناء إصدار للنواة . فإذا حدث ذلك، يرجى عمل تقرير لهذه الأخطاء وإرساله إلى مطوري نواة لينكس حتى يمكنهم إصلاحها.

خيارات متقدمة لبناء النواة :

نظام بناء النواة يتيح لك القيام بالعديد من الأشياء أكثر من مجرد بناء كامل للنواة وال modules.. الفصل العاشر يتضمن قائمة كاملة من الخيارات التي يوفرها نظام بناء النواة.

وفي هذا القسم، سوف نناقش بعض هذه الخيارات المتقدمة.

لمشاهدة وصف كامل لكيفية استخدام الخيارات المتقدمة الأخرى لبناء النواة، اذهب إلى الدليل *Documentation/kbuild* في الملف المصدري للنواة .

بناء أسرع للحاسبات متعددة المعالجات :

نظام بناء النواة يعمل بشكل جيد مع المهمة التي يمكنه تقسيمها إلى عدة قطع

(*)الإصدارات الأقدم من النواة 2.6 كانت تتطلب خطوة إضافية لبناء النواة وهي make modules لبناء كل kernel modules ولم تعد هناك حاجة الآن لهذه الخطوة .

مختلفة، وتوزيعها على عدة معالجات. وبذلك يمكنك استخدام الطاقة الكاملة من الحاسب متعدد المعالجات وتقليص الوقت اللازم لبناء النواة.
لبناء النواة بطريقة تعدد الخيوط multithreaded، استخدم الخيار *-j* مع برنامج *make*. ومن الأفضل إضافة رقم للخيار *-j* يقابل ضعف عدد المعالجات في النظام. لذلك بالنسبة للجهاز المحتوي على اثنين من المعالجات استخدم :

```
$ make -j4
```

وللجهاز الذي يحتوي على أربعة معالجات استخدم :

```
$ make -j8
```

إذا لم تمرر القيمة العددية للخيار *-j* :

```
$ make -j
```

سيقوم نظام بناء النواة بعمل خيط معالجة جديد لكل دليل فرعي في شجرة النواة، وذلك قد يؤدي إلى عدم استجابة جهازك وإمضاء الكثير من الوقت لاستكمال البناء.
ولذلك، فمن الموصى به أن تقوم دائما بتمرير قيمة عددية إلى الخيار *-j* .

بناء جزء من النواة فقط :

عند القيام بتطوير النواة أحيانا ترغب في بناء مجلد فرعي فقط مخصص، أو ملف مفرد داخل شجرة النواة. يتيح لك نظام بناء النواة فعل ذلك ببساطة . ولعمل بناء انتقائي لمجلد محدد، قم بتحديده من خلال سطر الأوامر، على سبيل المثال؛ لبناء الملفات الموجودة في المجلد *drivers/usb/serial* اكتب :

```
$ make drivers/usb/serial
```

على أية حال لن تقوم هذه الصيغة ببناء الصورة النهائية للموديول لهذا المجلد، وللقيام بذلك يمكنك استخدام الصيغة M= argument :

```
$ make M=drivers/usb/serial
```

والتي سوف تقوم ببناء كل الملفات اللازمة في هذا المجلد، وعمل رابط للصور النهائية للموديول .

وعندما تقوم ببناء مجلد مفرد بإحدى الطرق الموضحة سابقا، فإن الصورة النهائية للنواة لن يتم إعادة ربطها سوياً، ولذلك فإن التغييرات التي تم عملها للأدلة الفرعية لن تؤثر على الصورة النهائية للنواة، وذلك ربما لا يكون ما رغبت فيه ولذلك قم في النهاية بعمل :

```
$ make
```

لجعل نظام البناء يقوم بعمل فحص لجميع الملفات وعمل image link النهائية للنواة بشكل صحيح.

ولبناء ملف مخصص فقط في شجرة النواة، قم فقط بتمريره كقيمة للأمر *make*. على سبيل المثال، إذا كنت ترغب فقط في بناء موديل النواة */drivers/usb/serial/visor.ko*

اكتب :

```
$ make drivers/usb/serial/visor.ko
```

سيقوم نظام بناء النواة ببناء كل الملفات اللازمة لموديل النواة هذا visor.ko وعمل صورة للرابط النهائي لإنشاء هذا الموديل.

الملف المصدري في مكان والناتج في مكان آخر:

في بعض الأحيان ، يكون من السهل الحصول على الشفرة المصدرية لشجرة نواة في مكان للقراءة فقط (مثل قرص مدمج، أو في نظام تحكم لشفرة المصدر)، ومكان الخرج الناتج من بناء النواة في مكان آخر ،
بحيث أنك لا تشوش المصدر الأصلي لشجرة النواة. يقوم نظام بناء النواة بالتعامل مع ذلك بسهولة، ويشترط فقط القيمة O=option للإبلاغ عن المكان الذي يوضع فيه ناتج البناء. على سبيل المثال، إذا كان الملف المصدري للنواة يقع على قرص مدمج مربوط بالدليل */mnt/cdrom/* وكنت ترغب في وضع ملفات البناء في الدليل المحلي الخاص بك ، اكتب:

```
$ cd /mnt/cdrom/linux-2.6.17.11
make O=~/linux/linux-2.6.17.11
```

جميع ملفات البناء سيتم إنشاؤها في الدليل *~/linux/linux-2.6.17.11/* .
يرجى ملاحظة أن هذا الخيار O=option ينبغي أيضا تمريره إلى خيارات الإعداد الخاصة ببناء النواة، ولذلك فإنه يتم وضع ملفات الإعداد بشكل صحيح في دليل الناتج، وليس في الدليل الذي يحتوي على شفرة المصدر.

معماريات مختلفة Different Architectures

من أكثر الميزات المفيدة هي بناء النواة على شكل ترجمة متعدية أو هجينة cross-compiled لإتاحة المزيد من قوة الجهاز لبناء الكيرنل لنظام مضمن أصغر حجما، أو فقط لفحص البناء على معمارية معالج مختلفة، للتأكد من أن التغيير في الشفرة المصدرية لن يقوم بتدمير شيء ما بشكل غير متوقع.

نظام بناء الكيرنل يتيح لك تحديد معمارية مختلفة عن النظام الحالي عن طريق القيمة `ARCH= argument` .

ويسمح لك نظام البناء أيضا بتحديد مترجم مخصوص أنت ترغب فيه لعملية البناء باستخدام CC= argument ، أو(سلسلة أدوات) cross-compile toolchain مع CROSS_COMPILE argument

على سبيل المثال للحصول على إعدادات إفتراضية للنواة تعمل على معمارية المعالج x86_64 سوف تكتب :

```
$ make ARCH=x86_64 defconfig
```

لبناء جميع النواة على مجموعة أدوات ARM toolchain الموجودة في الدليل */usr/loca/bin/* عليك أن تكتب :

```
$ make ARCH=arm CROSS_COMPILE=/usr/local/bin/arm-linux-
```

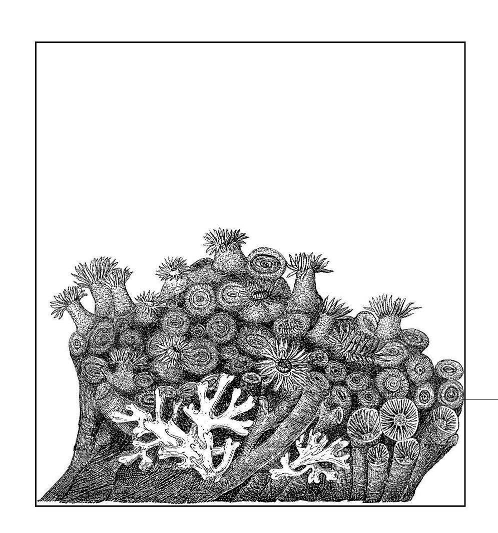
ومن المفيد أيضا للنواة غير المتعدية non-cross-compiled أن نغير ما يستخدمه نظام البناء للكومبايلر، ومن الأمثلة على ذلك استخدام البرامج *distcc* أو *ccache*، وكلاهما يساعد على تقليل الوقت المستغرق في بناء النواة .

لاستخدام برنامج *ccache* كجزء من نظام البناء اكتب :

```
$ make CC="ccache gcc"
```

ولاستخدام كلا البرنامجين *distcc* و *ccache* معا ،اكتب:

```
make CC="ccache distcc"
```





التثبيت والإقلاع من النواة

في الفصول السابقة بينا لك كيفية تحميل وبناء النواة الخاصة بك. والآن لديك ملف قابل للتنفيذ، إضافة إلى الوحدات modules التي قمت ببنائها، والآن حان الوقت لتثبيت النواة ومحاولة الإقلاع بها .

في هذا الفصل ، بعكس الفصول السابقة ، كل الأوامر تحتاج إلى المستخدم الجذر. ويمكن أن يتم هذا بوضع كلمة sudo في مقدمة كل أمر، أو عن طريق استخدام الأمر su لتصبح المستخدم الجذر، أو الدخول فعليا للنظام بصفة المستخدم الجذر. ولمعرفة ما إذا كان لديك *sudo* مثبتا ويعمل بشكل سليم، افعل ما يلي :

```
$ sudo ls ~/linux/linux-2.6.17.11/Makefile
Password:
Makefile
```

أدخل كلمة المرور الخاصة بك عند مؤشر كلمة :Password، أو كلمة السر الخاصة بمدير النظام (المستخدم الجذر). ويعتمد الخيار على كيفية تثبيت البرنامج sudo.إذا تم ذلك بنجاح ورأيت السطر يحتوي على ما يلي :

```
Makefile
```

، حينئذ يمكنك تخطي القسم التالي.

إذا لم يكن البرنامج *sudo* مثبتا، أو لا يعطيك الصلاحيات الصحيحة حاول استخدام الأمر *su* :

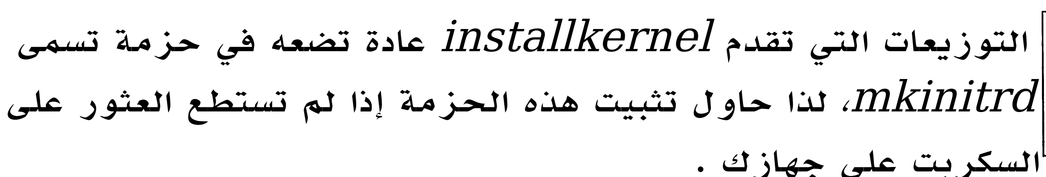
```
$ su
Password:
# exit
exit
$
```

عند المحث الخاص بكلمة المرور أدخل كلمة المرور لمدير النظام (*root*).
وعندما يقبل برنامج su كلمة المرور بنجاح، سوف يتم نقلك لإجراء أي شيء بالامتيازات الكاملة بالرووت.
كن شديد الحذر أثناء كونك root ، وقم فقط بعمل الحد الأدنى من احتياجاتك، وبعد ذلك قم بالخروج من البرنامج لمواصلة العمل بحساب مستخدم عادي.

استخدام سكربتات تثبيت التوزيعة

جميع التوزيعات تقريبا تأتي مع سكربت يسمى *installkernel* يمكن استخدامه من قبل نظام بناء النواة تلقائيا لتثبيت النواة المبنية في المكان الصحيح، وتعديل محمل الإقلاع bootloader، بحيث لا يوجد شيء زائد يقوم به المطور(*).





التوزيعات التي تقدم *installkernel* عادة تضعه في حزمة تسمى *mkinitrd*، لذا حاول تثبيت هذه الحزمة إذا لم تستطع العثور على السكربت على جهازك .

إذا قمت ببناء أي modules وتريد استخدام هذه الطريقة في تثبيت النواة اكتب أولا :

```
# make modules_install
```

وذلك سوف يبني كل ال modules التي قمت ببنائها، ويضعها في المكان المناسب في نظام الملفات كي تستطيع النواة الجديدة العثور عليها بشكل صحيح . توضع ال modules في المسار */lib/modules/kernel_version*، حيث إن *kernel_version* هو رقم إصدار النواة الجديدة التي قمت ببنائها آنفاً.
بعد أن يتم تثبيت ال modules بنجاح يجب تثبيت صورة النواة kernel image :

```
# make install
```

وهنا سوف تبدأ العمليات التالية :

1. نظام بناء النواة سيقوم بالتحقق من أن النواة تم بناؤها بنجاح.
2. سيقوم نظام بناء النواة بتثبيت الجزء الثابت من النواة داخل المجلد */boot* ويسمي هذا الملف التنفيذي بناء على إصدارة النواة التي تم بناؤها.

(*)ملحوظة: يستثنى من هذه القاعدة توزيعة Gentoo وغيرها من أنواع التوزيعات التي بنيت بنمط "من الصفر" "from scratch" والتي نتوقع من المستخدمين معرفة كيفية تركيب الأنوية على مسئوليتهم الشخصية. هذه الأنواع من التوزيعات تشتمل على وثائق عن كيفية تركيب نواة جديدة، و يتطلب ذلك أخذ المشورة لمعرفة الطريقة المطلوبة بالضبط.

3. أي صور أولية مطلوبة من قرص الذاكرة ramdisk يتم إنشاؤها تلقائيا ، باستخدام ال modules التي قمت بالفعل بتثبيتها أثناء مرحلة *modules_install* .
4. يتم إخطار برنامج محمل الإقلاع بوجود كيرنل جديد، وسوف يقوم بإضافته للقائمة المخصصةن ومن ثم يمكن للمستخدم اختيارها في المرة القادمة لإقلاع النظام.
5. وبعد الانتهاء من ذلك يكون قد تم تثبيت الكيرنل بنجاح، ويمكنك إعادة التشغيل بأمان وتجربة الكيرنل الجديد، فإذا كانت هناك أية مشكلة مع صورة الكيرنل الجديد يمكنك اختيار الكيرنل القديم وقت الإقلاع.

التثبيت بالطريقة اليدوية

إذا لم تكن توزيعتك تحتوي على الأمر *installkernel*، أو كنت ترغب فقط في عمل ذلك بيديك لتفهم خطوات معينة ، فهنا تجد مطلبك:

ال modules الواجب تثبيتها :

```
# make modules_install
```

الصورة الثابتة للنواة يجب نسخها إلى الدليل boot/ ،وللنواة المبنية على معمارية i386 اتبع الآتي:

```
# make kernelversion
2.6.17.11
```

لاحظ أن إصدار النواة ربما يكون مختلفا عما هو عندك ،

استخدم هذه القيمة مكان كلمة KERNEL_VERSION في الخطوات التالية :

```
# cp arch/i386/boot/bzImage /boot/bzImage-kERNEL_VERSION
# cp System.map /boot/System.map-KERNEL_VERSION
```

قم بالتعديل على محمل الإقلاع حتى يتعرف على الكيرنل الجديد.

ويشمل ذلك تعديل ملف ال configuration لمحمل الإقلاع الذي تستخدمه ، والذي سوف يتم تغطيته لاحقا في "تعديل محمل الإقلاع للنواة الجديدة" لمحمل الإقلاع من نوع GRUB و LILO.

إذا لم تعمل عملية الإقلاع بشكل سليم، فيكون ذلك عادة للحاجة إلى صورة أولية للقرص الذاكرة ramdisk.

ولعمل ذلك بشكل صحيح اتبع الخطوات التي ورد ذكرها في بداية هذا الفصل عن تثبيت الكيرنل أوتوماتيكياً،لأن سكربت تثبيت التوزيعة يعرف كيفية إنشاء قرص

الذاكرة-ramdisk [1] بشكل سليم مستخدما السكربتات والأدوت اللازمة. ولأن كل توزيعة تقوم بذلك بشكل مختلف، وذلك يتجاوز نطاق موضوع هذا الكتاب إلى تغطية جميع الأساليب المختلفة لبناء صورة قرص الذاكرة-ramdisk .

وها هنا سكربت بسيط يمكن استخدامه لتثبيت النواة أوتوماتيكيا بدلا من الحاجة إلى كتابة الأوامر السابقة طوال الوقت :

```
#!/bin/sh
#
# installs a kernel
#
make modules_install
# find out what kernel version this is
for TAG in VERSION PATCHLEVEL SUBLEVEL EXTRAVERSION ; do
    eval `sed -ne "/^$TAG/s/ //gp" Makefile`
done
SRC_RELEASE=$VERSION.$PATCHLEVEL.
$SUBLEVEL$EXTRAVERSION
# figure out the architecture
ARCH=`grep "CONFIG_ARCH " include/linux/autoconf.h | cut -f 2 -d
"\""`
# copy the kernel image
cp arch/$ARCH/boot/bzImage /boot/bzImage-"$SRC_RELEASE"
# copy the System.map file
cp System.map /boot/System.map-"$SRC_RELEASE"
echo "Installed $SRC_RELEASE for $ARCH"
```

تعديل محمل الإقلاع للنواة الجديدة

هناك نوعان شائعان من محمل الإقلاع GRUB و LILO،

و GRUB هو الأكثر شيوعا واستخداما في التوزيعات الحديثة، ويقوم بعمل بعض الأشياء أسهل قليلا مما يقوم به LILO، ولكن LILO مازال مشاهدا أيضا. وسوف نغطي كليهما في هذا الجزء.

للتحقق من نوع محمل الإقلاع الذي يستخدمه نظامك انظر في المجلد /boot إذا وجدت هناك مجلدا فرعيا باسم grub

---

(1) وهو نظام ملفات أولي يتم تحميله إلى ذاكرة الحاسوب بعد النواة ليساعدها على إكمال عملية الإقلاع بتوفير عدة ملفات منها مشغلات العتاد المستخدم على النظام.

$ ls -F /boot | grep grub

```
grub/
```

معنى ذلك أنك تستخدم برنامج grub للإقلاع، فإن لم يكن هذا المجلد موجودا ابحث عن الملف */etc/lilo.conf* :

$ ls /etc/lilo.conf

```
/etc/lilo.conf
```

إذا كان هذا الملف موجودا فأنت تستخدم برنامج LILO للقلاع . والخطوات التي تشتمل على إضافة نواة جديدة تختلف في برنامج منهما عن الآخر، لذلك اتبع فقط الجزء المناسب للبرنامج الذي تستخدمه.

GRUB

لتجعل GRUB يعلم بوجود كيرنل جديد، فكل ما تحتاجه هو تعديل الملف *./boot/grub/menu.lst*

لمعرفة كافة التفاصيل عن هذا الملف وجميع الخيارات المختلفة المتاحة، يرجى قراءة صفحات info الخاصة ب GRUB :

$ info grub

أسهل طريقة لإضافة كيرنل جديد داخل الملف */boot/grub/menu.lst* هو نسخ خانة موجودة بالفعل، على سبيل المثال تأمل ملف *menu.lst* الخاص بتوزيعة Gentoo :

```
timeout 300
default 0
splashimage=(hd0,0)/grub/splash.xpm.gz
title 2.6.16.11
    root (hd0,0)
    kernel /bzImage-2.6.16.11 root=/dev/sda2 vga=0x0305
title 2.6.16
    root (hd0,0)
    kernel /bzImage-2.6.16 root=/dev/sda2 vga=0x0305
```

The line starting with the word title defines a new kernel entry, so this file
contains two entries. Simply copy one block of lines beginning with the title line,

السطر يبدأ بكلمة title يعرف خانة النواة الجديدة ، لذلك يحتوي هذا الملف على

خانتين. قم ببساطة بنسخ كتلة واحدة من الأسطر البادئة بسطر title كما يلي :

```
title 2.6.16.11
        root (hd0,0)
        kernel /bzImage-2.6.16.11 root=/dev/sda2 vga=0x0305
```

بعد ذلك أضف الكتلة لنهاية الملف، وقم بتحرير رقم الإصدار ليحتوي رقم إصدار النواة التي انتهيت من تثبيتها. الاسم title ليس مهما، طالما هو فريد من نوعه، ولكنه يعرض في قائمة الإقلاع، لذا ينبغي أن تجعل له معنى مفيدا، وفي مثالنا قمنا بتثبيت الكيرنل 2.6.17.11، لذا النسخة النهائية ستكون شبيهة بالآتي:

```
timeout 300
default 0
splashimage=(hd0,0)/grub/splash.xpm.gz
title 2.6.16.11
    root (hd0,0)
    kernel /bzImage-2.6.16.11 root=/dev/sda2 vga=0x0305
title 2.6.16
    root (hd0,0)
    kernel /bzImage-2.6.16 root=/dev/sda2 vga=0x0305
title 2.6.17.11
    root (hd0,0)
    kernel /bzImage-2.6.17.11 root=/dev/sda2 vga=0x0305
```

بعد أن تقوم بحفظ الملف، أعد التشغيل النظام، وكن متأكدا أن اسم صورة الكيرنل الجديد واردة في قائمة الإقلاع.

استخدم السهم الأسفل لتحديد الكيرنل الجديد ثم اضغط مفتاح Enter للإقلاع بصورة الكيرنل الجديد.

LILO

لتجعل LILO يعلم بوجود الكيرنل الجديد، يجب عليك أن تقوم بتعديل الملف /etc/lilo.conf وبعد ذلك شغل الأمر *lilo* لتفعيل ما تم من تغييرات لملف الإعداد.

لمعرفة التفاصيل الكاملة عن هيكلية ملف إعداد LILO يرجى الاطلاع على الصفحات

الدليلية manpage الخاصة ب LILO.

$ man lilo

أسهل طريقة لإضافة كيرنل جديد داخل الملف */etc/lilo.conf* هو نسخ مادة موجودة بالفعل، على سبيل المثال تأمل ملف إعداد LILO الخاص بنظام(توزيعة) Gentoo :

```
boot=/dev/hda
prompt
timeout=50
default=2.6.12
image=/boot/bzImage-2.6.15
   label=2.6.15
   read-only
   root=/dev/hda2
image=/boot/bzImage-2.6.12
   label=2.6.12
   read-only
   root=/dev/hda2
```

السطر الذي يبتدئ بالعبارة =image يشير لمدخل النواة الجديدة، لذا يحتوي هذا الملف على اثنين من الخانات. قم ببساطة بنسخ كتلة واحدة من الأسطر مع السطر =image، كما يلي :

```
image=/boot/bzImage-2.6.15
   label=2.6.15
   read-only
   root=/dev/hda2
```

بعد ذلك أضف هذه الكتلة إلى نهاية الملف، واكتب رقم الإصدار الذي يحتوي عليه النواة التي قمت بتثبيتها آنفاً.

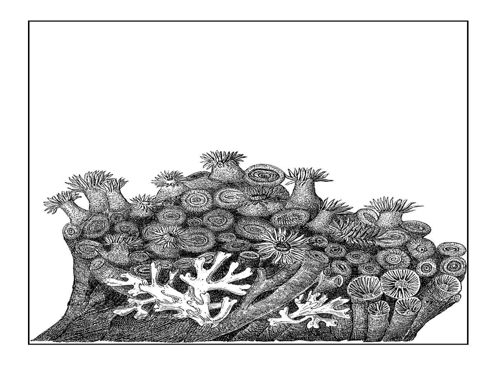
الاسم title ليس مهما، طالما أنه فريد من نوعه، ولكنه يعرض في قائمة الإقلاع، لذا ينبغي أن تجعل له معنى مفيدا، وفي مثالنا قمنا بتثبيت الكيرنل 2.6.17.11، لذا النسخة النهائية ستكون شبيهة بالآتي:

```
boot=/dev/hda
prompt
timeout=50
default=2.6.12
image=/boot/bzImage-2.6.15
    label=2.6.15
    read-only
    root=/dev/hda2
image=/boot/bzImage-2.6.12
    label=2.6.12
    read-only
    root=/dev/hda2
image=/boot/bzImage-2.6.17
    label=2.6.17
    read-only
    root=/dev/hda2
```

بعد حفظك للملف شغل البرنامج */sbin/lilo/* لكتابة التعديلات على الإعدادات لقسم الإقلاع على القرص الصلب :

```
# /sbin/lilo
```

والآن يمكن إعادة تشغيل النظام بأمان. خيار النواة الجديدة يمكن رؤيته في قائمة الأنوية المتاحة وقت الإقلاع. استخدم السهم الأسفل لتحديد النواة الجديدة، ثم اضغط مفتاح Enter للإقلاع بصورة النواة الجديدة.



6

ترقية النواة

حتما يحدث الآتي : لديك نواة مبنية ومخصصة، وهي تعمل بشكل رائع باستثناء شيء واحد بسيط وأنت تعلم أنه تم إصلاحه في أحدث إصدار من قبل مطوري النواة . أو وجدت لديك مشكلة أمنية، وهناك إصدار لنواة جديدة مستقرة تم الإعلان عنها. وفي كلتا الحالتين، فإنك تواجه مسألة ترقية النواة ،وفي نفس الوقت لا تريد أن تخسر الوقت والجهد الذين أنفقتهما لعمل تهيئة متقنة للنواة .

هذا الفصل يريك بسهولة كيفية تحديث النواة من نسخ أقدم ،بينما تظل جميع خيارات تهيئة النواة القديمة تعمل.

أولا يرجى عمل نسخة احتياطية لملف التهيئة config. في مجلد الملف المصدري الخاص بالنواة.وسوف تمضي بعض الوقت والجهد في إنشائه،ويجب حفظه خوفا من حدوث أي خطأ أثناء محاولة عملية الترقية .

```
$ cd ~/linux/linux-2.6.17.11
$ cp .config ../good_config
```

هناك خمس خطوات فقط لازمة لعمل ترقية للنواة من النواة القديمة:

1. الحصول على الملف المصدري للشفرة.
2. تطبيق التغييرات على شجرة الملف المصدري القديم لترقيته للمستوى الأحدث.
3. إعادة تهيئة النواة على أساس إعدادات النواة السابقة.
4. بناء النواة الجديدة.
5. تثبيت النواة الجديدة.

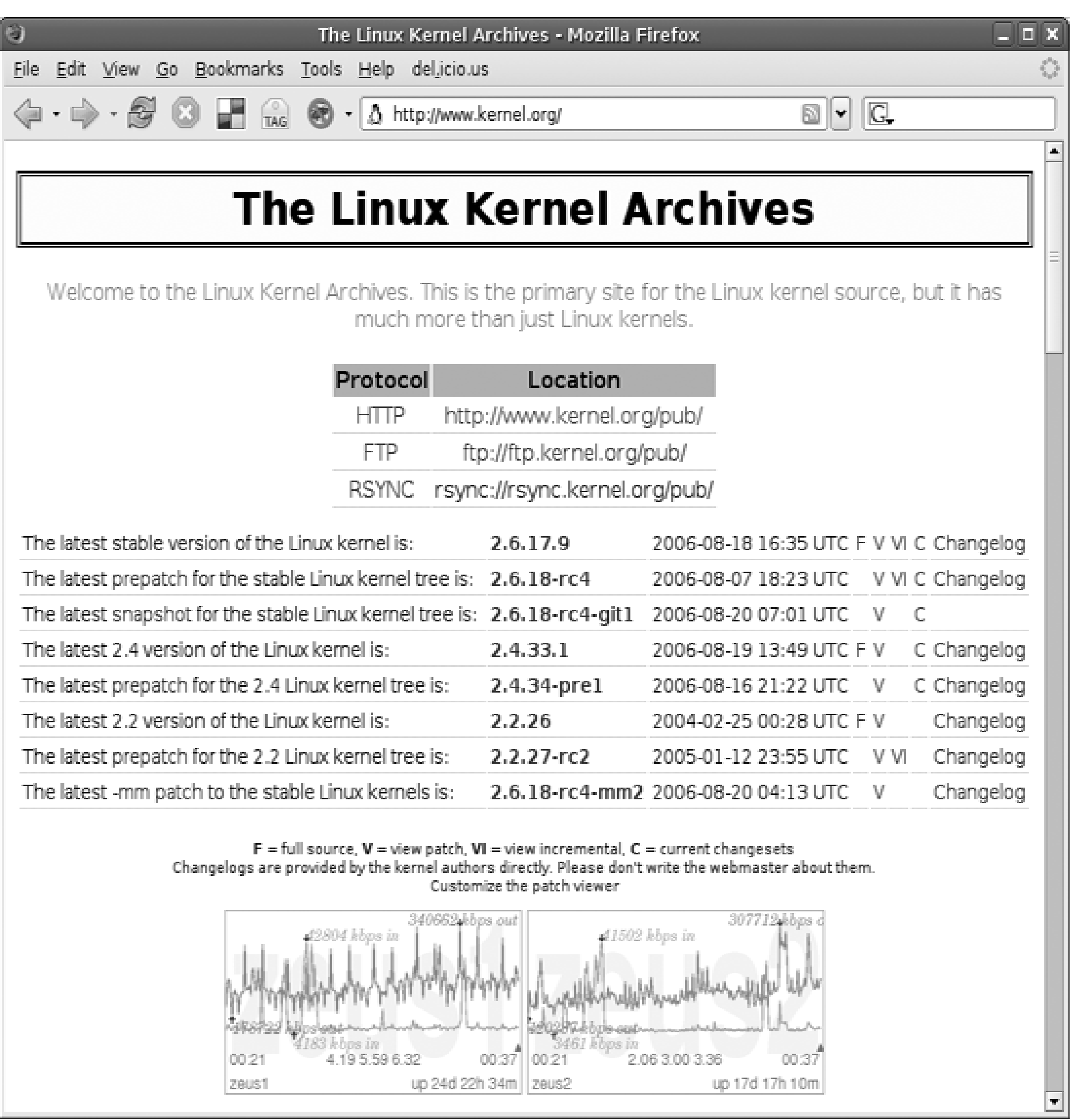
آخر خطوتين تعملان بنفس الشكل الذي تم شرحه من قبل،لذلك سوف نقوم فقط بمناقشة الثلاث خطوات الأولى في هذا الفصل.

في هذا الفصل نفترض أنك قمت ببناء إصدار النواة 2.6.17.9 بنجاح وتريد

الترقية إلى الإصدار 2.6.17.11.

تحميل الملف المصدري الجديد:

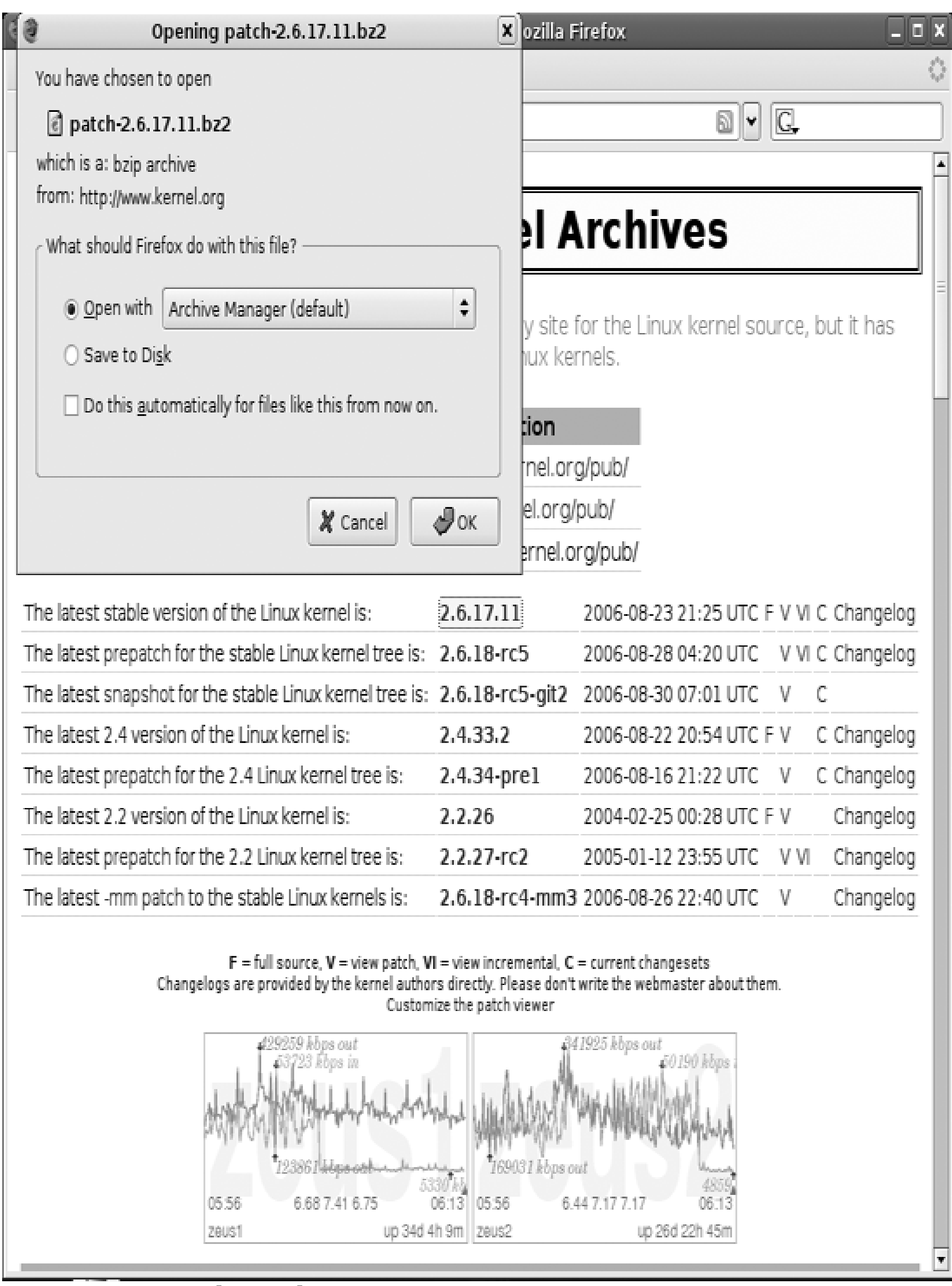
أدرك مطورو نواة لينكس أن المستخدمين لا يرغبون في تحميل كامل لشفرة المصدرمع كل تحديث. حيث سيكون ذلك مضيعة لل bandwidth والوقت. وبسبب هذا ، قاموا بتقديم باتش (*) يمكنه ترقية النواة القديمة إلى إصدار أحدث. على الصفحة الرئيسيةل kernel.org على شبكة الانترنت ، سوف تتذكر أنه يتضمن قائمة لنسخ النواة الحالية المتاحة للتحميل، كما هو مبين في الشكل 6-1.



الشكل 6-1:الموقع الرئيسي لـ kernel.org

(*) وهو يسمى باتش (أي الدفعة) لأن برنامج الباتش ياخذ الملف ويطبقه على شجرة الكيرنل الأصلي منشئا شجرة جديدة. يحتوي ملف الباتش على تمثيل للتغيرات الضرورية لإعادة بناء الملفات على أساس الملفات القديمة .ملفات الباتش قابلة للقراءة وتحتوي على قائمة من الأسطر التي ستحذف والأسطر التي ستضاف،مع بعض القرائن بداخل الملف توضح مكان حدوث هذه التغييرات.

سابقا ، استخدمت الوصلة المشار إليها بالحرف F لتحميل كامل الشفرة المصدرية للنواة. بينما، إذا نقرت على اسم إصدار النواة، سوف يتم تحميل ملف الباتش بدلا من ذلك ، كما هو مبين في الشكل 6-2.



الشكل 6-2:تحميل الباتش من موقع kernel.org

وذلك ما نريد عمله عند الترقية.ولكننا نريد معرفة ما هو الباتش الذي علينا تحميله.

ما الباتش الذي ينطبق على الإصدار؟

سيقوم ملف باتش النواة بترقية الشفرة المصدرية فقط من إصدار محدد إلى إصدار محدد آخر.وهنا نبين كيفية استعمال ملفات الباتش المختلفة :

- باتشات النواة المستقرة مخصصة لنسخة النواة الرئيسية.ذلك معناه أن الباتش 2.6.17.10 مخصص فقط لإصدار النواة رقم 2.6.17 .ولا يصلح باتش النواة 2.6.17.10 للنواة 2.6.17.9 ولا لأي إصدار آخر.
- باتشات النواة الرئيسية مخصصة فقط لنسخ النواة الرئيسية السابقة، ذلك معناه أن باتش النواة 2.6.17.18 يصلح فقط للإصدار 2.6.17، ولا يصلح للنواة y.2.6.17 أو أي إصدار آخر.
- الباتشات التصاعدية incremental patches تقوم بالترقية من إصدار نواة معين للإصدار الذي يليه.وذلك يتيح للمطورين عدم التراجع بالنواةثم ترقيتها بعد ذلك.يجب فقط التحول من آخر إصدار لنواة مستقرة إلى الإصدار المستقر التالي(تذكر أن باتشات الإصدار المستقر من النواة يكون موجها فقط للإصدار الرئيسي،وليس للإصدار المستقر السابق له) .وكلما كان ذلك ممكنا ، فمن الموصى استخدام الباتشات التصاعدية incremental patches لجعل حياتك أسهل.

إيجاد الباتش :

إذا أردنا الترقية من إصدارة النواة 2.6.17.9 إلى الإصدار 2.6.17.11 فيجب علينا تحميل باتشين مختلفين. سوف نحتاج إلى باتش للترقية من الإصدار 2.6.17.9 إلى الإصدار 2.6.17.10،وبعد ذلك الترقية من الإصدار 2.6.17.10 إلى الإصدار 2.6.17.11 (*) .

باتشات النواة الرئيسية والمستقرة تكون موجودة في نفس الدليل الرئيسي لشجرة ملف المصدر.

كل ال incremental patches يمكن العثور عليها بالنزول درجة واحدة لأسفل في الدليل الفرعي incr. وذلك للحصول على الباتش الذي يرقي من 2.6.17.9 إلى 2.6.17.10 . نبحث في الدليل */linux/kernel/v2.6/incr*

(*)إذا كنت تريد الترقية بمقدار أكثر من نسختين، فمن الموصى به كطريقة لتوفير الخطوات أن ترجع للخلف، وبعد ذلك تقوم بالترقية للأمام، وفي هذه الحالة علينا أن نرجع للوراء من النسخة 2.6.17.9 إلى 2.6.17 ، وبعدها نتقدم للأمام من الإصدار 2.6.17 إلى 2.6.17.11 .

للعثور على الملفات التي نريد[**]:

```
$ cd ~/linux
     $ lftp ftp.kernel.org/pub/linux/kernel/v2.6/incr
cd ok, cwd=/pub/linux/kernel/v2.6/incr
ftp ftp.kernel.org:/pub/linux/kernel/v2.6/incr> ls
*2.6.17.9*.bz2
-rw-rw-r--    1 536      536          2872 Aug 22 19:23
patch-2.6.17.9-10.
bz2
lftp ftp.kernel.org:/pub/linux/kernel/v2.6/incr> get
patch-2.6.17.9-10.bz2
2872 bytes transferred
lftp ftp.kernel.org:/pub/linux/kernel/v2.6/incr> get
patch-2.6.17.10-11.bz2
7901 bytes transferred
lftp ftp.kernel.org:/pub/linux/kernel/v2.6/incr> exit
$ ls -F
good_config linux-2.6.17.9/ patch-2.6.17.10-11.bz2
patch-2.6.17.9-10.bz2
```

تطبيق الباتش:

ولأن الباتشات التي تم تحميلها مضغوطة، فيجب علينا أولا أن نفك ضغطها بالأمر : *bzip2*

```
$ bzip2 -dv patch-2.6.17.9-10.bz2
 patch-2.6.17.9-10.bz2: done
$ bzip2 -dv patch-2.6.17.10-11.bz2
 patch-2.6.17.10-11.bz2: done
$ ls -F
good_config linux-2.6.17.9/ patch-2.6.17.10-11
patch-2.6.17.9-10
```

والآن نحتاج إلى تطبيق ملفات هذا الباتش داخل دليل النواة ،فاذهب إلى هذا الدليل:

```
$ cd linux-2.6.17.9
```

والآن قم بتشغيل برنامج الباتش لتنقل شجرة الشفرة المصدرية من الإصدار 2.6.17.9 إلى الإصدار 2.6.17.10 :

```
$ patch -p1 < ../patch-2.6.17.9-10
patching file Makefile
```

(**)في هذا المثال : استخدمنا برنامج ممتاز لنقل الملفات lftp FTP لتحميل ملفات الباتش .اي برنامج FTP أو متصفح إنترنت يمكنك استخدامه لتحميل نفس الملفات .والمهم هو معرفة مكان نزول الملفات.

```
patching file block/elevator.c
patching file fs/udf/super.c
patching file fs/udf/truncate.c
patching file include/net/sctp/sctp.h
patching file include/net/sctp/sm.h
patching file net/sctp/sm_make_chunk.c
patching file net/sctp/sm_statefuns.c
patching file net/sctp/socket.c
```

تأكد بأن الباتش يعمل فعلا بشكل صحيح، وأنه ليس هناك ثمت أخطاء أو تحذيرات في الخرج-output الخاص ببرنامج الباتش .

وإنها لفكرة جيدة أيضا أن تنظر في *Makefile* الخاص بالنواة لترى رقم إصدارة النواة :

```
$ head -n 5 Makefile
VERSION = 2
PATCHLEVEL = 6
SUBLEVEL = 17
EXTRAVERSION = .10
NAME=Crazed Snow-Weasel
```

الآن وقد أصبحت النواة في مستوى الإصدار 2.6.17.10،قم بعمل نفس الأمر كما سبق وقم بتطبيق الباتش لترقية النواة إلى الإصدار 2.6.17.11 :

```
$ patch -p1 < ../patch-2.6.17.10-11
patching file Makefile
patching file arch/ia64/kernel/sys_ia64.c
patching file arch/sparc/kernel/sys_sparc.c
patching file arch/sparc64/kernel/sys_sparc.c
patching file drivers/char/tpm/tpm_tis.c
patching file drivers/ieee1394/ohci1394.c
patching file drivers/md/dm-mpath.c
patching file drivers/md/raid1.c
patching file drivers/net/sky2.c
patching file drivers/pci/quirks.c
patching file drivers/serial/Kconfig
patching file fs/befs/linuxvfs.c
patching file fs/ext3/super.c
patching file include/asm-generic/mman.h
patching file include/asm-ia64/mman.h
patching file include/asm-sparc/mman.h
```

```
patching file include/asm-sparc64/mman.h
patching file kernel/timer.c
patching file lib/spinlock_debug.c
patching file mm/mmap.c
patching file mm/swapfile.c
patching file net/bridge/netfilter/ebt_ulog.c
patching file net/core/dst.c
patching file net/core/rtnetlink.c
patching file net/ipv4/fib_semantics.c
patching file net/ipv4/netfilter/arp_tables.c
patching file net/ipv4/netfilter/ip_tables.c
patching file net/ipv4/netfilter/ipt_ULOG.c
patching file net/ipv4/route.c
patching file net/ipx/af_ipx.c
patching file net/netfilter/nfnetlink_log.c
```

قم مرة أخرى بالتحقق من أن الخرج الخاص ببرنامج الباتش لا يظهر أية أخطاء وانظر في ملف *Makefile* :

```
$ head -n 5 Makefile
VERSION = 2
PATCHLEVEL = 6
SUBLEVEL = 17
EXTRAVERSION = .11
NAME=Crazed Snow-Weasel
```

الآن تم تحديث الشفرة المصدرية بنجاح إلى النسخة التي ترغب في استخدامها، وإنها لفكرة جيدة أن تعود لتغيير اسم الدليل ليشير إلى رقم إصدارة النواة الجديدة كي تتجنب أي تعارض في المستقبل :

```
$ cd ..
    $ mv linux-2.6.17.9 linux-2.6.17.11
    $ ls -F
    good_config linux-2.6.17.11/ patch-2.6.17.10-11 patch-2.6.17.9-10
```

إعادة تهيئة النواة :

في السابق ، استخدمنا طريقة *make menuconfig* أو *gconfig* أو *xconfig* لتغيير الخيارات المختلفة للتهيئة. ولكن بمجرد حصولك على خيارات تهيئة عاملة ، فإنه من الضروري أن تقوم بتحديثها بالخيارات الجديدة تم إضافتها للنواة بعد آخر إصدار. وللقيام بذلك يجب استخدام خيارات *make oldconfig*

و *make*، و *silentoldconfig*.

يقوم *make oldconfig* بأخذ خيارات التهيئة للنواة الحالية في ملف *config*. ويقوم بتحديثه على أساس إصدارة النواة الجديدة. ولفعل ذلك قم بكتابة كل الأسئلة الخاصة بالتهيئة، وقم بالإجابة عليها إذا كان الخيار مستخدما بالفعل في ملف التهيئة. فإذا كان هناك خيار جديد سيتوقف البرنامج ويسأل المستخدم ما هي القيمة التي يجب وضعها لخيار التهيئة الجديد .بعد الإجابة أمام المحث يواصل البرنامج العمل حتى تنتهي إعدادات تهيئة النواة بالكامل.

الأداة *make silentoldconfig* تعمل تماما مثل طريقة *oldconfig*، ولكنها لا تقوم بطباعة أي شيء على الشاشة إلا إذا احتاجت إلى السؤال عن خيار جديد للتهيئة.

وعادة عند عمل ترقية بين نسخ مختلفة من الإصدارات المستقرة للنواة، لا يوجد خيارات تهيئة جديدة يتم إضافتها لأنه من المفترض أنها سلسلة لنواة مستقرة.

وإذا حدث ذلك فليس هناك اسئلة جديدة مطلوب الإجابة عنها لتهيئة النواة، لذلك يواصل البرنامج عمله بنجاح دون حاجة لأي تدخل من المستخدم .

ومثال على ذلك الترقية من الإصدار 2.6.17.9 إلى الإصدار 2.6.17.11 :

```
$ cd linux-2.6.17.11
$ make silentoldconfig
scripts/kconfig/conf -s arch/i386/Kconfig
#
# using defaults found in .config
#
```

المثال التالي يوضح ما يحدث عندما تظهر خيارات جديدة لنواة جديدة.

خيار النواة الذي علينا تفعيله *Mutex debugging* يعتبر جديدا على النواة الحالية. وهنا نرى الخرج عندما يحدث ذلك :

```
$ make silentoldconfig
 scripts/kconfig/conf -s arch/i386/Kconfig
 #
 # using defaults found in .config
 #
 *
 * Restart config...
 *
 *
 * Kernel hacking
 *
```

Show timing information on printks (PRINTK_TIME) [Y/n/?] y
Magic SysRq key (MAGIC_SYSRQ) [Y/n/?] y
Kernel debugging (DEBUG_KERNEL) [Y/n/?] y
Kernel log buffer size (16 => 64KB, 17 => 128KB)
(LOG_BUF_SHIFT) [16] 16
Detect Soft Lockups (DETECT_SOFTLOCKUP) [Y/n/?] y
Collect scheduler statistics (SCHEDSTATS) [N/y/?] n
Debug slab memory allocations (DEBUG_SLAB) [Y/n/?] y
Memory leak debugging (DEBUG_SLAB_LEAK) [Y/n] y
Mutex debugging, deadlock detection (DEBUG_MUTEXES) [N/
y/?] (NEW) y

The configuration program stops at this option and asks for the user to choose an
option. Press y, and the program continues on:

Spinlock debugging (DEBUG_SPINLOCK) [Y/n/?] y
Sleep-inside-spinlock checking (DEBUG_SPINLOCK_SLEEP) [Y/
n/?] y
kobject debugging (DEBUG_KOBJECT) [N/y/?] n
Highmem debugging (DEBUG_HIGHMEM) [N/y/?] n
Compile the kernel with debug info (DEBUG_INFO) [N/y/?] n
Debug Filesystem (DEBUG_FS) [Y/?] y
Debug VM (DEBUG_VM) [N/y/?] n
Compile the kernel with frame pointers (FRAME_POINTER)
[N/y/?] n
Compile the kernel with frame unwind information
(UNWIND_INFO) [N/y/?] n
Force gcc to inline functions marked 'inline'
(FORCED_INLINING) [N/y/?] n
torture tests for RCU (RCU_TORTURE_TEST) [N/m/y/?] n
Check for stack overflows (DEBUG_STACKOVERFLOW) [N/y/?] n
Stack utilization instrumentation (DEBUG_STACK_USAGE)
[N/y/?] n
Stack backtraces per line (STACK_BACKTRACE_COLS) [2] 2
*
* Page alloc debug is incompatible with Software Suspend on
i386
*
Write protect kernel read-only data structures
(DEBUG_RODATA) [N/y/?] n

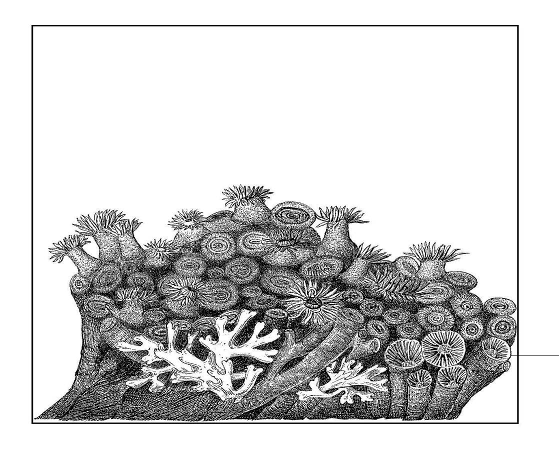
Use 4Kb for kernel stacks instead of 8Kb (4KSTACKS) [N/y/?] n

ولذا فإن ترقية تهيئة النواة لإصدار جديد بسيط كما لو أنك تستخدم خيار تهيئة مختلف ل *make*.

مع هذه الطريقة أنت لا تحتاج لاستخدام برامج رسومية أو نصية للتهيئة من أجل أي تحديث جديد للنواة.

ألا يمكن عمل ذلك آليــا ؟

كل العمليات بدءا من تحميل ملف الباتش المناسب، وفك الضغط، و التنفيذ تبدو أنها مهيئة للأتمتة.ويبدو أن مطوري النواة من الطراز الذي يحب أتمتة المهام المكررة، وقد تم إنشاء برنامج *ketchup* للتعامل مع كل ذلك آليا. انظر الملحق A لمزيد من التفاصيل عن كيفية عمل هذا البرنامج وكيفية استخدامه.



7

تخصيص النواة

إن من أصعب الأجزاء في عملية بنائك لنسختك الخاصة من النواة هي أن تحدد بالضبط المشغلات وخيارات التهيئة اللازمة لعمل جهازك على نحو سليم.
هذا الفصل سيمضي بك خلال عملية إيجاد وانتقاء هذه المشغلات.

استخدام نواة التوزيعة :

من أسهل الطرق لتحديد ما هي ال modules الضرورية، هو البدء من خلال إعدادات النواة التي تأتي مع حزم توزيعتك. والأسهل كذلك في تحديد المشغلات اللازمة لتشغيل النظام، حيث توجد المشغلات الملائمة للعتاد ومحددة بالفعل.
إذا لم يكن هناك توزيعة مثبتة بالفعل على الجهاز الذي تبني له النواة؛ استخدم نسخة أسطوانة حية LiveCD لإحدى التوزيعات.وذلك يسمح بإقلاع لينكس على الجهاز وتحديد ما تحتاجه النواة من خيارات التهيئة لجعل عتاد الجهاز يعمل بشكل سليم .

أين توجد إعدادات النواة ؟

كل التوزيعات تزود النواة بملفات تهيئة كجزء من حزم النواة الخاصة بالتوزيعة.
قم بقراءة الوثائق الخاصة بالنواة لمعرفة كيفية العثور على هذه الإعدادات. وعادة ما توجد تحت المسار */usr/src/linux/* .
إذا كان العثور على ملفات التهيئة أمرا صعبا ابحث في مجلد النواة نفسها. أغلب أنوية التوزيعات مبنية مع إضافة التهيئة بداخل نظام الملفات /proc .
للتحقق من وجود ذلك على نواتك العاملة اكتب :

```
$ ls /proc/config.gz
/proc/config.gz
```

إذا كان الملف /proc/config.gz موجودا فقم بنسخه إلى دليل الملف المصدري لنواتك وفك ضغطه كما يلي :

```
$ cp /proc/config.gz ~/linux/
$ cd ~/linux
$ gzip -dv config.gz
    config.gz:        74.9% -- replaced with config
```

قم بنسخ ملف التهيئة هذا إلى مجلد ال kernel لديك وأعد تسميته إلى *.config* .

بعد ذلك قم باستخدامه كأساس لتهيئة وبناء النواة كما سبق شرحه في الفصل الرابع.

استخدام هذا الملف للتهيئة يتولد عنه صورة للنواة للعمل على جهازك.

أقل ميزة لصورة هذه النواة هو أنك قمت تقريبا ببناء كل موديل نواة ومشغل موجود في شجرة الملف المصدري للنواة .وهذا ما لا نحتاجه في أغلب الأحوال لجهاز واحد، لذا يمكنك البدء في تعطيل بعض المشغلات والخيارات المختلفة التي لا تحتاجها.

ومن الموصى به أن تقوم بتعطيل تلك الخيارات التي أنت متيقن من عدم حاجتك لها فقط، فقد يكون هناك أجزاء من النظام تحتاج بالفعل لتلك الخيارات المحددة التي تم تفعيلها.

إيجاد الموديل الذي تحتاج إليه :

ملف التهيئة الذي يأتي مع التوزيعة يستغرق الكثير من الوقت لبنائه، وذلك لأن جميع مشغلات الأجهزة يتم بناؤها معه.ولكنك تريد بناء مشغلات العتاد الذي لديك فحسب، وذلك سيوفر وقت بناء النواة، ويتيح لك أيضا بناء بعض أو كل هذه المشغلات بداخل النواة نفسها، ومن الممكن توفير بعض الذاكرة، وعلى بعض معماريات المعالجات تعمل بنظام أكثر سرعة.

ولتنزيل مشغلات العتاد الخاص بك يجب عليك أن تحدد ال modules اللازمة لتشغيل عتادك .

وسوف نعمل هنا من خلال مثالين عن كيفية اكتشاف المشغلات اللازمة للتحكم في قطع العتاد .

هناك العديد من المواضع على نظامك تختزن معلومات مفيدة لتحديد الأجهزة وما

يرتبط بها من مشغلات على النواة العاملة الآن .

أهم هذه المواضع هو نظام الملفات الافتراضي المسمى *sysfs*.

ويجب دائما ربط *sysfs*. مع المسار */sys/* في نظام الملفات لديك، وذلك بواسطة السكربتات الأولية لتوزيعة لينكس الخاصة بك.

Sysfs يزودك بلمحات عن كيفية العمل المتناغم بين أجزاء النواة، مع الروابط الرمزية symlinks [1] الدالة على كل أنحاء نظام الملفات.

في كل الأمثلة التالية ترى مسارات فعلية ل *sysfs* وأنواع من العتاد .ولعل عتاد جهاز مختلف ولكن أماكن المعلومات ستكون هي ذاتها.

لا تشعر بالقلق إذا كانت أسماء الملفات في *sysfs* مختلفة عن جهازك، فإن ذلك أمر متوقع.

وبالإضافة إلى ذلك، فإن الهيكل الداخلي نظام الملفات *sysfs* يتغير باستمرار ويرجع ذلك إلى إعادة تنظيم الأجهزة وإعادة التفكير من قبل مطوري النواة) عن أفضل السبل لعرض الهياكل الداخلية للنواة و userspace. بسبب هذا ،و مع مرور الوقت، فإن بعض ال symlinks التي سبق ذكرها في هذا الفصل قد لا تكون موجودة. ومع ذلك، فإن المعلومات كلها لا تزال هناك، فقط ابحث حولها قليلا.

مثال : التحقق من مشغل جهاز الشبكة :

إن من أهم الأجهزة الشائعة في النظام هو بطاقة الشبكة، ومن الضروري معرفة المشغل المتحكم في هذا الجهاز وتفعيله في ملف تهيئة النواة الخاص بك، حتى تعمل الشبكة بشكل سليم.

أولا ، خذ خلفية عن اسم الاتصال الشبكي لاكتشاف جهاز PCI المتحكم بها .

---

(1)Symlinks هو اختصار لـ(SYmbolic LinK (SYLK ويسمى أيضا soft link ، وهو نوع خاص من الملفات يحتوي على رابط أو إشارة لمسار خاص بملف أو مجلد معين وهو شبيه باختصارات وندوز windows shortcuts (والحقيقة أن وندوز مقلد له) ويتاثر الرابط بكل تغير يحدث داخل الملف الأصلي، فإذا ألغي الأصل أو تغير اسمه فلا فائدة من الرابط، حيث إنه يشير إلى ملف غير موجود ، بينما إذا ألغي الرابط فلن يتاثر الملف الاصلي.

وقد ظهر لأول مرة من خلال نظام (BSD (Berkeley Software Distributionسنة 1983 ثم انتقل بعد ذلك لأنظمة Unix-like operating systems مثل لينوكس وماك وأوبن بي إس دي وسولاريس ثم إلى وندوز فيستا

و هذا الرابط تستفيد منه البرامج بحيث تستخدمه في القراءة والكتابة إلى الملفات والمجلدات عن طريق هذا الرابط .

ولعمل ذلك انظر في الأسماء المختلفة للشبكة :

```
$ ls /sys/class/net/
eth0 eth1 eth2 lo
```

الدليل *lo* يمثل جهاز شبكة loopback، وليس ملحقا بأي جهاز شبكةحقيقي. الأدلة *eth0* و *eth1,* و *eth2* هي ما أريدك أن تنتبه إليه، حيث إنها تمثل بطاقات شبكة حقيقية .

لمزيد من البحث في أجهزة الشبكة هذه لمعرفة ما يهمك منها، استخدم الأداة *ifconfig* :

```
$ /sbin/ifconfig -a
eth0      Link encap:Ethernet HWaddr 00:12:3F:65:7D:C2
          inet addr:192.168.0.13 Bcast:192.168.0.255
Mask:255.255.255.0
          UP BROADCAST NOTRAILERS RUNNING MULTICAST MTU:1500
Metric:1
          RX packets:2720792 errors:0 dropped:0 overruns:0
frame:0
          TX packets:1815488 errors:0 dropped:0 overruns:0
carrier:0
          collisions:0 txqueuelen:100
          RX bytes:3103826486 (2960.0 Mb) TX bytes:371424066
(354.2 Mb)
          Base address:0xdcc0 Memory:dfee0000-dff00000
eth1      Link encap:UNSPEC HWaddr 80-65-00-12-7D-
C2-3F-00-00-00-00-00-00-
          00-00-00
          BROADCAST MULTICAST MTU:1500 Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0
carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b) TX bytes:0 (0.0 b)
eth2      Link encap:UNSPEC HWaddr 00-02-3C-04-11-09-D2-
BA-00-00-00-00-00-
          00-00-00
          BROADCAST MULTICAST MTU:1500 Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0
carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b) TX bytes:0 (0.0 b)
lo        Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          UP LOOPBACK RUNNING MTU:16436 Metric:1
          RX packets:60 errors:0 dropped:0 overruns:0 frame:0
```

```
          TX packets:60 errors:0 dropped:0 overruns:0
carrier:0
          collisions:0 txqueuelen:0
      RX bytes:13409 (13.0 Kb) TX bytes:13409 (13.0 Kb)
```

من خلال تلك القائمة يمكنك الإبلاغ بأن البطاقة eth0 هي جهاز الشبكة النشط والذي يعمل، كما ترى في هذه الأسطر :

```
eth0 Link encap:Ethernet HWaddr 00:12:3F:65:7D:C2
     inet addr:192.168.0.13 Bcast:192.168.0.255
Mask:255.255.255.0
```

يوضح هذا الناتج أن هناك جهاز إيثرنت مع عنوان IP صالح يشير إليه .
والآن قد تحققنا من أن الجهاز eth0 سوف يعمل على النواة الجديدة ،ونحتاج العثور على المشغل المتحكم بذلك الجهاز.وذلك ببساطة بالتحرك إلى روابط مختلفة في نظام الملفات *sysfs* والذي يمكن عمله بسطر أوامر واحد :

```
$ basename `readlink /sys/class/net/eth0/device/driver/module`
e1000
```

يظهر الناتج أن اسم الموديل e1000 يتحكم في جهاز الشبكة eth0 .والأمر *basename* يتضح أنه اختصر الخطوات التالية في سطر واحد من الأوامر :

1. اتبع الرابط */sys/class/net/eth0/device* داخل شجرة الدليل */sys/device* والذي يحتوي على معلومات عن الجهاز المتحكم في eth0. ولاحظ أن الدليل */sys/class/net/eth0* ربما يكون أيضا عبارة عن symlink (رابط)في النسخ الحديثة من النواة.
2. بداخل الدليل الذي يصف الجهاز في *sysfs* ، يوجد symlink (رابط) للمشغل يشير لهذا الجهاز. هذا ال symlink يسمى *driver*، لذا سنتبع هذا الرابط.
3. بداخل الدليل الذي يصف مشغل الجهاز في *sysfs* ، يوجد symlink للمشغل يشير للموديل الذي يتضمنه هذ االمشغل بداخله. هذا ال symlink يسمى module .ونحن نريد الملف الأصلي لهذا الرابط.وللحصول على الأصل استخدمنا الأمر *readlink*، والذي يعطينا ناتجا يشبه التالي :

```
$ readlink /sys/class/net/eth0/device/driver/module
../../../../module/e1000
```

4. ولأننا نهتم فقط باسم الموديل نريد اقتطاع بقية شريط المسار الناتج من الأمر *readlink*، وحفظ الجزء الأيمن منه فقط .وهذا ما يقوم به الأمر . *basename*

وتنفيذ الأمر على اسم الملف مباشرة سوف ينتج :

```
$ basename ../../../../module/e1000
```

e1000

ثم نضع ناتج ال symlink الطويل الموصل إلى مكان *readlink* داخل برنامج *basename* ، نفعل العملية برمتها من خلال سطر أوامر واحد.

الآن وقد حصلنا على اسم الموديل علينا أن نعثر على خيار تهيئة النواة المتحكم به. يمكنك البحث في القوائم المختلفة لإعدادات جهاز الشبكة، أو البحث في الملف المصدري للنواة نفسها للتأكد من أنك اخترت الخيار الصحيح.

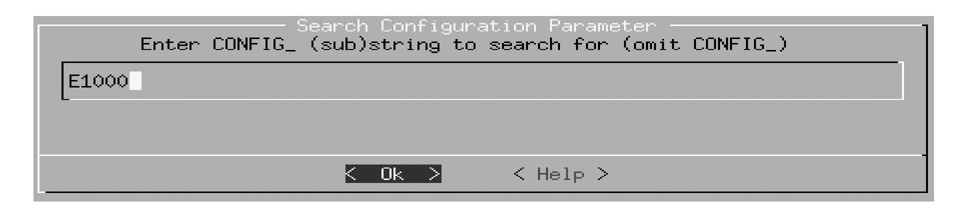
```
$ cd ~/linux/linux-2.6.17.8
$ find -type f -name Makefile | xargs grep e1000
./drivers/net/Makefile:obj-$(CONFIG_E1000) += e1000/
./drivers/net/e1000/Makefile:obj-$(CONFIG_E1000) += e1000.o
./drivers/net/e1000/Makefile:e1000-objs := e1000_main.o
e1000_hw.o e1000_
ethtool.o e1000_param.o
```

لا تنس أن تستبدل e1000 المستخدم في هذا المثال باسم الموديل الذي تبحث عنه. الأمر المهم الذي ينبغي أن تنظر إليه في ناتج الأمر *find* السابق هو السطر الذي يحتوي على المصطلح _CONFIG .حيث إن ذلك هو خيار التهيئة الذي تحتاج النواة إلى تفعيله لبناء الموديل.وفي المثال السابق يعتبر الخيار CONFIG_E1000 هو خيار التهيئة الذي تبحث عنه.

والآن لديك المعلومات اللازمة لتهيئة النواة. قم بتشغيل قائمة أدوات التهيئة التالية:

```
$ make menuconfig
```

بعدها اضغط المفتاح / (والذي يبادر بالبحث) واكتب _CONFIG .وهذه العملية تظهر في الشكل 7-1.



الشكل 7-1:البحث في menuconfig

سيقوم نظام بناء النواة بعدها باخبارك بالضبط عن المكان الذي تفعّل فيه هذا الموديل. انظر الشكل 7-2.

```
Search Results
Symbol: E1000 [=y]
Prompt: Intel(R) PRO/1000 Gigabit Ethernet support
  Defined at drivers/net/Kconfig:1901
  Depends on: NET && !UML && PCI
  Location:
    -> Device Drivers
      -> Network device support
        -> Ethernet (1000 Mbit)


Symbol: E1000_DISABLE_PACKET_SPLIT [=n]
Prompt: Disable Packet Split for PCI express adapters
  Defined at drivers/net/Kconfig:1940
  Depends on: NET && !UML && E1000
  Location:
    -> Device Drivers
                                                              ( 50%)
                          < Exit >
```

الشكل 7-

2:نتيجة البحث في menuconfig

أول نتيجة معروضة تطابق بالضبط ما تبحث عنه.والمعلومات المعروضة تخبرك بأنه يجب عليك بناء الموديل E1000 داخل النواة،وأنه يجب تفعيل خيار التهيئة التالي:

```
Device Drivers
  Network device support
        [*] Network device support
            Ethernet (1000 Mbit)
      [*] Intel(R) PRO/1000 Gigabit Ethernet support
```

هذه الخطوات سوف تعمل مع أي نوع من الأجهزة العاملة داخل النواة.

مثال: جهاز USB

مثال آخر، دعنا ننظر في المحول من USB-إلى-serial الذي يوجد كمثال في نظامنا. وهو متصل الآن بالمدخل */dev/ttyUSB0/* لذا فأنت تحتاج للنظر في القسم *sysfs tty* :

```
$ ls /sys/class/tty/ | grep USB
ttyUSB0
```

يمكنك من خلال *sysfs* تعقب هذا الجهاز لإيجاد الموديل المتحكم به، كما يتضح في القسم السابق :

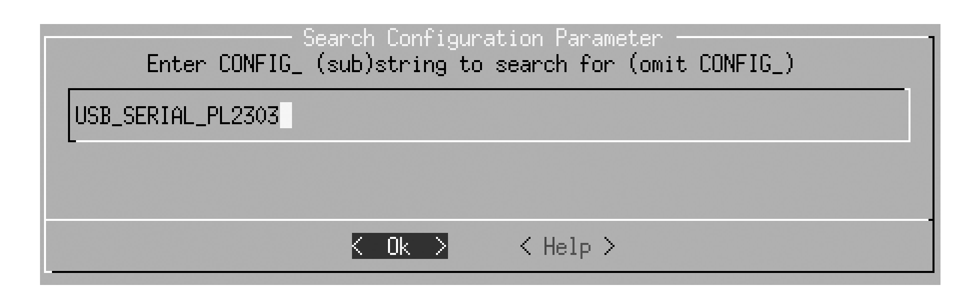
```
$ basename `readlink /sys/class/tty/ttyUSB0/device/driver/module`
pl2303
```

بعد ذلك قم بالبحث داخل شجرة الملف المصدري للنواة للعثور على خيار التهيئة الذي تحتاج لتفعيله:

```
$ cd ~/linux/linux-2.6.17.8
```

```
$ find -type f -name Makefile | xargs grep pl2303
./drivers/usb/serial/Makefile:obj-$(CONFIG_USB_SERIAL_PL2303)
+= pl2303.o
```

استخدم أداة تهيئة النواة كما يظهر في الشكل 7-3 لإيجاد الخيار المناسب الذي يجب تفعيله لإعداد الخيار CONFIG_USB_SERIAL_PL2303.



الشكل 7-3 : البحث عن USB_SERIAL_PL2303

وفي حالتنا يعرض لنا الشاشة التي تظهر في الشكل 7-4.حيث تعرض بالضبط المكان الذي يوجد فيه الخيار USB Prolific 2303 Single Port Serial Driver واللازم للتحكم بهذا الجهاز على نحو سليم.

```
                    Search Results
Symbol: USB_SERIAL_PL2303 [=m]
Prompt: USB Prolific 2303 Single Port Serial Driver
  Defined at drivers/usb/serial/Kconfig:442
  Depends on: USB!=n && USB_SERIAL
  Location:
    -> Device Drivers
      -> USB support
        -> USB Serial Converter support
          -> USB Serial Converter support (USB_SERIAL [=m])

                                                    (100%)
                    < Exit >
```

شكل 7-4: نتيجة البحث عن USB_SERIAL_PL2303

ملخص في استكشاف الأجهزة

في هذا الملخص توجد الخطوات اللازمة للعثور على مشغل أحد الأجهزة الذي يمتلك بالفعل مشغلا عاملا ومرتبطا به.

1. أوجد ال class device *sysfs* المرتبط بالجهاز. ستجد أجهزة الشبكة في قائمة داخل */sys/class/net* وأجهزة tty في */sys/class/tty/* والأنواع الأخرى من الأجهزة في مسارات أخرى داخل */sys/class/*، ويعتمد ذلك على

نوع الجهاز.

2. تتبع شجرة الملفات داخل *sysfs* لإيجاد اسم الموديل الذي يتحكم بذلك الجهاز.وسوف تجده في المسار
/sys/class/class_name/device_name/device/driver/ module
ويمكن عرضه باستخدام البرنامجين: *readlink* و *basename* :

```
$ basename `readlink
/sys/class/class_name/device_name/device/driver/module`
```

3. ابحث باستخدام *Makefiles* لإيجاد *CONFIG_ rule* التي تبنى اسم هذا الموديل باستخدام الأمر *find* والأمر *grep*:

```
$ find -type f -name Makefile | xargs grep module_name
```

4. ابحث في نظام تهيئة النواة عن قيمة التهيئة هذه، واذهب إلى الموضع في القائمة الذي يحدده لتفعيل ذلك المشغل حتى يتسنى بناؤه.

دع النواة تخبرنا بما نحتاج :

الآن وقد انتهينا من جميع الخطوات من البحث حول *sysfs* ثم تعقب روابط *symlinks* أسماء الموديلز، نقدم هنا سكربت غاية في البساطة، يقوم بكل هذا العمل بطريقة مختلفة :

```
#!/bin/bash
#
# find_all_modules.sh
#
for i in `find /sys/ -name modalias -exec cat {} \;`; do
    /sbin/modprobe --config /dev/null --show-depends $i ;
done | rev | cut -f 1 -d '/' | rev | sort -u
```

يمكنك تحميل ملف لأحد الأمثلة التي تحتوي على هذا السكربت من الموقع الإلكتروني للكتب المذكورة في قسم "كيف تتصل بنا " “ How to Contact Us” في افتتاحية هذا الكتاب.

ويسعى هذا السكربت خلال *sysfs*، ويجد جميع الملفات المسماة *modalias*.
وملف *modalias* هذا يتضمن اسم الموديل الذي يخبر الأمر *modprobe* بالموديل الذي يجب تحميله للتحكم بهذا الجهاز . هذا الموديل يتكون من مزيج من

اسم مصنع الجهاز ، و ID ، ونوع الصنف ، وغيرها من الأوصاف الفريدة التي تحدد نوع الجهاز. و جميع modules المشغلات في النواة لها قائمة داخلية من الأجهزة التي تدعمها والتي تتولد تلقائيا من خلال قائمة الأجهزة. والمشغل يخبر النواة بما يدعمه.

يقوم *modprobe* بالبحث خلال هذه القائمة من الأجهزة عن كل المشغلات K ويحاول التوفيق بينها وبين الاسم الذي تحمله. فإذا وجده مطابقا فسوف يقوم بتحميل الموديل (وذلك الإجراء يبين كيف يتم التحميل التلقائي للمشغل أثناء عمل لينكس).

يتوقف السكربت الذي يحمل برنامج *modprobe* عن العمل قبل تحميل الموديل بالفعل، ويقوم فقط بطباعة الأحداث التي سيجريها. ويعطينا قائمةمن ال modules اللازمة للتحكم بكافة الأجهزة على النظام.

وبقليل من التنقية للقائمة عن طريق ترتيبها وإيجاد الحقل المناسب لعرض نتائجه في هذا الخرج :

```
$ find_all_modules.sh
8139cp.ko
8139too.ko
ehci-hcd.ko
firmware_class.ko
i2c-i801.ko
ieee80211.ko
ieee80211_crypt.ko
ipw2200.ko
mii.ko
mmc_core.ko
pcmcia_core.ko
rsrc_nonstatic.ko
sdhci.ko
snd-hda-codec.ko
snd-hda-intel.ko
snd-page-alloc.ko
snd-pcm.ko
snd-timer.ko
snd.ko
soundcore.ko
uhci-hcd.ko
usbcore.ko
yenta_socket.ko
```

هذه قائمة بكل ال modules اللازمة للتحكم بالعتاد في هذا الجهاز.

ومن المحتمل أن يقوم السكربت أيضا بطباعة بعض رسائل الخطأ شبيهة بما يلي :

```
FATAL: Module
pci:v00008086d00002592sv000010CFsd000012E2bc03sc00i00 not
found.
FATAL: Module serio:ty01pr00id00ex00 not found.
```

وهذا معناه أنه لم يستطع إيجاد الموديل القادر على التحكم بهذا الجهاز.ولا تقلق تجاه ذلك حيث إن بعض الأجهزة ليس لها مشغل في النواة يعمل معها.

تحديد الموديل الصحيح منذ البداية:

أحيانا لا يكون لديك الخيار للحصول على نواة لتوزيعة تعمل على الجهاز كي تتحقق من أنواع modules اللازمة لتشغيل العتاد.

أو أنك قمت بإضافة قطع جديدة من العتاد إلى نظامك، وتحتاج إلى معرفة ما هي خيارات تهيئة النواة اللازم تفعيلها لتعمل هذه القطع بشكل سليم .

هذا القسم سوف يساعدك لتحدد كيف تجد خيار التهيئة لتنصيب الجهاز ثم تشغيله.

وأسهل طريقة لمعرفة المشغل الذي يتحكم بالجهاز الجديد هو بناء جميع المشغلات المختلفة لهذا النوع بداخل شجرة مصدر النواة على أنه modules، وترك *udev* يبدأ عملية مطابقة المشغل بالجهاز.

وبمجرد حدوث ذلك، ستكون قادرا على العمل بالعودة إلى الوراء، مستخدما الخطوات التي تم مناقشتها لتحديد المشغل الصحيح المطلوب، وبعدها ارجع وقم فقط بتفعيل المشغل في ملف تهيئة النواة.

ولكن إذا لم تكن ترغب في بناء كل المشغلات، أو أنها لا تعمل لسبب ما، فسوف يتطلب ذلك قليلا من الجهد لتحديد المشغل المناسب المطلوب.

الخطوات التاليةمعقدة وتتطلب التعمق في الشفرة المصدرية للنواة لوقت طويل.

لا تخف من ذلك سيساعدك ذلك فقط لتفهم عتادك والشفرة المصدرية للنواة جيدا.

الخطوات التي تنطوي عليها مطابقة المشغل مع الجهاز تختلف اعتمادا على نوع هذا الجهاز الذي تعمل عليه. وسوف نناقش نوعين من أكثر الأنواع شيوعا من الأجهزة في هذا الفصل وهما :أجهزة PCI وأجهزة USB. والطرق الموصوفة هنا سوف تعمل أيضا مع أنواع أخرى من الأجهزة.

أيضا ، من المهم جدا للنواة أن تتمكن من إيجاد جميع نظم الملفات في النظام ، وأهمها نظام ملفات الجذر. وسنبين كيفية القيام بذلك في وقت لاحق في "نظام ملفات الجذر".

أجهزة PCI :

أجهزة PCI تتميز بهوية للبائع vendor ID ، وهوية للجهاز device ID؛ الجمع بين كل من هوية البائع وهوية الجهاز يتطلب مشغلا فريد من نوعه. وهذا هو أساس البحث الذي يوضحه لك هذا الجزء.
في هذا المثال ، دعونا نستخدم بطاقة شبكة من نوع PCI، وهي لا تعمل حاليا مع نسخة النواة العاملة. هذا المثال سيكون مختلفا عن حالتك، مع جهاز PCI آخر وقيم مختلفة لل ID، ولكن ينبغي أن تكون الخطوات التي تنطوي عليها ذات صلة بأي نوع من أجهزة PCI ترغب في إيجاد مشغل يعمل معه.
أولا أوجد جهاز PCI لا يعمل على النظام .وللحصول على قائمة من كل أجهزة PCI استخدم برنامج *lspci*، ولأننا نهتم فقط ببطاقة إيثرنت PCI فسوف نقترب ببحثنا عن أجهزة PCI بالبحث فقط بعبارة تحتوي على مصطلح Ethernet (مع حساسية لحالة الحرف):

```
$ /usr/sbin/lspci | grep -i ethernet
06:04.0 Ethernet controller: Realtek Semiconductor Co., Ltd.
RTL-8193/8139C/8139C+ (rev 10)
```

هذا هو الجهاز الذي نرغب في تشغيله.(*)

كل التوزيعات تقريبا تضع برنامج *lspci* في المسار */usr/sbin/* ولكن البعض منها يضعه في موضع آخر.ولإيجاد مكان البرنامج اكتب :

```
$ which lspci
/usr/sbin/lspci
```



إذا كنت تستخدم توزيعة تضع البرنامج في مكان ما يرجى استخدام هذا المسار كلما ناقشنا استخدام الأمر *lspci*.
أول أجزاء من ناتج الأمر *lspci* تبين PCI bus ID لهذا الجهاز،06:04.0. هذه هي القيمة التي سنستخدمها عند البحث خلال *sysfs* لاكتشاف مزيد من المعلومات عن ذلك الجهاز.
اذهب إلى *sysfs* حيث توجد جميع الأنواع المختلفة من أجهزة PCI في قائمة، وقم بالنظر في أسمائها:

(*)لاحظ أنه يمكنك فقط محاولة البحث من خلال ملف تهيئة النواة عن جهاز مطابق للعبارة الموضحة هنا ، وهو جهاز من Realtek Semiconductor مع اسم المنتج وهو
RTL-8193/8139C/8139C + ، ولكن هذا لا يعمل دائما . وهذا هو السبب في اتخاذنا الطريق الطويل في هذا الفصل.

```
$ cd /sys/bus/pci/devices/
$ ls
0000:00:00.0 0000:00:1d.0  0000:00:1e.0 0000:00:1f.3 0000:06:03.3
0000:00:02.0   0000:00:1d.1  0000:00:1f.0 0000:06:03.0 0000:06:03.4
0000:00:02.1  0000:00:1d.2  0000:00:1f.1 0000:06:03.1 0000:06:04.0
0000:00:1b.0 0000:00:1d.7  0000:00:1f.2 0000:06:03.2 0000:06:05.0
```

ترقم النواة أجهزة PCI بالبادئة 0000: والتي لا تظهر في ناتج البرنامج *lspci* (**) لذا قم بإضافة البادئة 0000:أمام الرقم الذي تجده عند استخدام *lspci* ثم اذهب لهذا المسار

```
$ cd 0000:06:04.0
```

في هذا المسار تحتاج أن تعرف القيم الخاصة باسماء ملفات ال *vendor* و *device* :

```
$ cat vendor
0x10ec
$ cat device
0x8139
```

يوجد هناك الأرقام التعريفية للبائع ولجهاز الـ PCI. وتستخدم النواة هذه القيم لتقرن بين التعريف وبين الجهاز بشكل سليم .تقوم مشغلات أجهزة PCI بإبلاغ النواة بالـ ID الخاص بالبائع والجهاز الذي تدعمه لذلك تتعرف النواة على كيفية الربط بين المشغل وبين الجهاز المناسب له.

قم بكتابة ذلك في مكان ما حيث إننا سنشير إليه في وقت لاحق.

الآن وقد عرفنا id البائع والمنتج لجهاز PCI ، نحن نحتاج للعثور على المشغل الذي يعلن أن النواة تدعم الجهاز الخاص به.ارجع إلى الدليل الخاص بالملف المصدري للنواة:

```
$ cd ~/linux/linux-2.6.17.8/
```

المكان الشائع وجود معرفات أجهزة PCI فيه داخل شجرة الملف المصدري للنواة هو المسار */linux/pci_ids.h/.*

ابحث عن الملف الخاص برقم بائع المنتج vendor product number:

```
$ grep -i 0x10ec include/linux/pci_ids.h
#define PCI_VENDOR_ID_REALTEK           0x10ec
```

القيمة المعرفة هنا PCI_VENDOR_ID_REALTEK هي ما يحتمل استخدامه في أي مشغل للنواة يهدف إلى دعم اجهزة من هذا المصنع.

(**) بعض المعالجات من فئة 64 بت سوف تظهر leading bus number لأجهزة PCI في ناتج الأمر lspci ولكن الأغلب الأعم من أجهزة لينوكس لا تظهر هذا الرقم افتراضيا.

ولتكون في مأمن قم أيضا بالنظر في هذا الملف عن الرقم التعريفي (ID)لجهازنا، والذي تم وصفه هناك أيضا :

```
$ grep -i 0x8139 include/linux/pci_ids.h
#define PCI_DEVICE_ID_REALTEK_8139      0x8139
```

هذا التعريف سينفعنا في وقت لاحق.

والآن ابحث عن الملفات المصدرية للمشغل التي تشير إلى ID هذا البائع :

```
$ grep -Rl PCI_VENDOR_ID_REALTEK *
include/linux/pci_ids.h
drivers/net/r8169.c
drivers/net/8139too.c
drivers/net/8139cp.c
```

لسنا في حاجة إلى إلقاء نظرة على أول الملف المدرج بالقائمة هنا ، pci_ids.h ، لأن هذا هو المكان الذي وجدنا به التعريف الأصلي. ولكن الملفات ، *r8139.c* ، *8139too.c* ، و *8169cp.c* في الدليل الفرعي */drivers/net* ينبغي أن تدرس عن قرب أكثر.

افتح واحدا من هذه الملفات بأي محرر نصي وابحث عن : *PCI_VENDOR_ID_REALTEK*. في ملف *drivers/net/r8169.c* ، وهو يبدو بوضوح في هذا الجزء من الشفرة :

```
static struct pci_device_id rtl8169_pci_tbl[] = {
        { PCI_DEVICE(PCI_VENDOR_ID_REALTEK, 0x8169), },
        { PCI_DEVICE(PCI_VENDOR_ID_REALTEK, 0x8129), },
        { PCI_DEVICE(PCI_VENDOR_ID_DLINK,   0x4300), },
        { PCI_DEVICE(0x16ec,                0x0116), },
        { PCI_VENDOR_ID_LINKSYS,            0x1032,
PCI_ANY_ID, 0x0024, },
        {0,},
};
```

جميع مشغلات أجهزة pci تحتوي على قائمة لمختلف الأجهزة التي تدعمها.

هذه القائمة واردة في الهيكل الخاص بقيم pci_device_id. وذلك يشبه هذا المثال.وهذا ما نحتاج للنظر فيه لتحديد ما هو الجهاز المدعوم من هذا المشغل .والقيمة الخاصة بالبائع (vendor) مطابقة هنا، ولكن القيمة الثانية بعد اسم البائع هي القيمة الخاصة بالجهاز.والجهاز الخاص بنا يحمل القيمة 0x8169 و 0x8129 للأجهزة ذات الرقم التعريفي(id)للبائع الخاصة بـ PCI_VENDOR_ID_REALTEK ، ولذلك فإن هذا المشغل لن يدعم جهازنا.

انتقل إلى الملف التالي drivers/net/8139too.c، نجد عبارة

في الجزء التالي من الشفرة : PCI_VENDOR_ID_REALTEK

```
if (pdev ->vendor == PCI_VENDOR_ID__REALTEK &&
    pdev->device == PCI_DEVICE_ID_REALTEK_8139 && pci_rev >=
0x20) {
    dev_info(&pdev->dev,
           "This (id %04x:%04x rev %02x) is an enhanced
8139C+ chip\n",
           pdev->vendor, pdev->device, pci_rev);
    dev_info(&pdev->dev,
           "Use the \"8139cp\" driver for improved
performance and
stability.\n");
}
```

استخدام قيمة PCI_VENDOR_ID_REALTEK هنا أيضا يتطابق مع الشفرة التي تقوم بفحص ما إذا كان الرقم التعريفي لجهاز PCI مطابق للقيمة PCI_DEVICE_ID_REALTEK_8139 .

إذا حدث ذلك يقوم المشغل بطباعة رسالة تقول: "استخدم المشغل 8139cp لتحسين الأداء والاستقرار." " Use the 8139cp driver for improved performance and stability."

ربما ينبغي علينا النظر في ذلك المشغل فيما بعد. حتى ولو لم يكن لدينا مثل هذا الدليل الواضح ، فالمشغل 8139too.c لا يحمل رقم هوية البائع ولا رقم هوية الجهاز الذي نبحث عنه في صيغة المتغير pci_ device_id، وذلك يعطينا الدليل أنه لن يدعم جهازنا.

وفي النهاية، ابحث في الملف drivers/net/8139cp.c. فهو يستخدم التعريف PCI_VENDOR_ID_REALTE في هذا الجزء من الشفرة:

```
static struct pci_device_id cp_pci_tbl[] = {
        { PCI_VENDOR_ID_REALTEK, PCI_DEVICE_ID_REALTEK_8139,
          PCI_ANY_ID, PCI_ANY_ID, 0, 0, },
        { PCI_VENDOR_ID_TTTECH, PCI_DEVICE_ID_TTTECH_MC322,
          PCI_ANY_ID, PCI_ANY_ID, 0, 0, },
        { },
};
MODULE_DEVICE_TABLE(pci, cp_pci_tbl);
```

هنا يستخدم القيم الخاصة بكل من رقم هوية البائع ورقم هوية الجهاز في صيغة اسم المتغير pci_device_id. هذا المشغل ينبغي أن يدعم جهازنا .

والآن وقد حصلنا على اسم المشغل، يمكننا العودة للوراء كما هو موضح في الجزء الأول من هذا الفصل، للحصول على القيمة الخاصة بتهيئة النواة التي يجب تفعيلها لبناء هذا المشغل.

وفي هذا الملخص، يوجد هنا الخطوات اللازمة للعثور على مشغل PCI الذي يمكنه التحكم بجهاز PCI معين :

1. أوجد PCI bus ID الخاص بالجهاز الذي تريد العثور على المشغل الخاص به، باستخدام الأمر *lspci*.
2. اذهب إلى الدليل */sys/bus/pci/devices/0000:bus_id*،حيث إن *bus_id* هو PCI bus ID الموجود بالخطوة السابقة.
3. اقرأ القيم الخاصة بملفات البائع والجهاز في الدليل الخاص بجهاز PCI .
4. ارجع للوراء لشجرة الملف المصدري للنواة، واطلع على *include/linux/pci_ids*. لمعرفة أرقام الهوية-IDs الخاصة بأجهزة PCI والموجودة في الخطوة السابقة.
5. ابحث في شجرة الملف المصدري للنواة عما يشيرلقيم هذه المشغلات .كلا من ID الخاص بالبائع والجهاز يجب أن يكون في صيغة تعريف pci_device_id.
6. ابحث في Makefiles النواة عن CONFIG_ rule الذي يبني هذا المشغل باستخدام الأمرين *find* و *grep*:

```
$ find -type f -name Makefile | xargs grep DRIVER_NAME
```

7. ابحث في نظام تهيئة النواة عن قيمة التهيئة هذه، واذهب إلى المكان المحدد بالقائمة لتفعيل المشغل ليجري بناؤه .

أجهزة usb

العثور على المشغل المحدد لجهاز USB يشبه إلى حد كبير العثور على مشغل جهاز ل PCI كم تم شرحه في القسم السابق، مع اختلافات طفيفة في قيم bus ID .

في هذا المثال دعنا نعثر على المشغل اللازم لتشغيل جهاز وايرلس USB. وكما حدث مع المثال الخاص بجهاز PCI ،ستكون التفاصيل في هذا المثال مختلفة عما لديك، ولكن الخطوات التي تنطوي عليها ينبغي أن تكون ذات صلة بأي نوع من أجهزة USB التي ترغب في العثور على مشغل يعمل عليها.

وكما حدث مع جهاز PCI، ينبغي أن يوجد bus ID لجهاز usb الذي ترغب في إيجاد مشغل له. وللقيام بذلك يمكنك استخدام برنامج *lsusb* الذي يأتي مع الحزمة *usbutils*.

برنامج *lsusb* يعرض كل أجهزة USB الملحقة بنظامك، وحيث إنك لا تعرف بماذا يسمى الجهازالمحدد الذي تبحث عنه ، ابدأ بالنظر في جميع أجهزة usb:

```
$ /usr/sbin/lsusb
Bus 002 Device 003: ID 045e:0023 Microsoft Corp. Trackball
Optical
Bus 002 Device 001: ID 0000:0000
Bus 005 Device 003: ID 0409:0058 NEC Corp. HighSpeed Hub
Bus 005 Device 001: ID 0000:0000
Bus 004 Device 003: ID 157e:300d
Bus 004 Device 002: ID 045e:001c Microsoft Corp.
Bus 004 Device 001: ID 0000:0000
Bus 003 Device 001: ID 0000:0000
Bus 001 Device 001: ID 0000:0000
```

الجهاز الذي يحمل الهوية رقم 0000:0000 يمكن تجاهله، وكذلك المتحكم المضيف الذي يشغل الناقل نفسه،وبترشيحهم يتركنا مع أربعة أجهزة فقط :

```
$ /usr/sbin/lsusb | grep -v 0000:0000
    Bus 002 Device 003: ID 045e:0023 Microsoft Corp.
    Trackball Optical
    Bus 005 Device 003: ID 0409:0058 NEC Corp. HighSpeed Hub
    Bus 004 Device 003: ID 157e:300d
    Bus 004 Device 002: ID 045e:001c Microsoft Corp.
```

وبما أن أجهزة USB يسهل نزعها، قم بنزع الجهاز الذي تريد مشغل له، وشغل الأمر *lsusb* مرة أخرى:

```
$ /usr/sbin/lsusb | grep -v 0000:0000
Bus 002 Device 003: ID 045e:0023 Microsoft Corp. Trackball
Optical
Bus 005 Device 003: ID 0409:0058 NEC Corp. HighSpeed Hub
Bus 004 Device 002: ID 045e:001c Microsoft Corp.
```

الجهاز الثالث الآن غير موجود ، مما يعني أن هذ الجهاز :Bus 004 Device 003: ID 157e:300d هو الجهاز الذي تريد إيجاد مشغل له .

إذا قمت باستبدال الجهاز ثم النظر في ناتج الأمر *lsusb* مرة أخرى سوف يتغير رقم الجهاز :

```
$ /usr/sbin/lsusb | grep 157e
    Bus 004 Device 004: ID 157e:300d
```

وذلك لأن أرقام جهاز USB غير فريدة في نوعها، ولكنه يتغير في كل مرة يتم توصيله بالجهاز.والأمر الثابت هو id الخاص بالبائع المنتج، والذي يظهر هنا من خلال الأمر *lsusb* على شكل قيمتين كل منها مكونة من أربع خانات مع نقطتين ":" بين كل منها.

وبالنسبة لهذا الجهاز، ID البائع هو 157e و ID المنتج هو 300d. قم بتسجيل هذه القيم التي وجدتها حيث إنك ستستخدمها في الخطوات القادمة.

ومثلما حدث مع جهاز PCI سنقوم بالبحث في الملف المصدري للنواة عن ID البائع

والمنتج الخاصة بجهاز USB حتى نجد المشغل المناسب للتحكم بهذا الجهاز.ولسوء الحظ لا يوجد ملف واحد يحتوى على كل أرقام الهوية IDs الخاصة بأجهزة USB ، مثلما هو الحال مع أجهزة PCI، ولذلك فمن الضروري البحث في شجرة الملف المصدري للنواة كلها :

$ grep -i -R -l 157e drivers/*

```
drivers/atm/pca200e.data
drivers/atm/pca200e_ecd.data
drivers/atm/sba200e_ecd.data
drivers/net/wireless/zd1211rw/zd_usb.c
drivers/scsi/ql1040_fw.h
drivers/scsi/ql1280_fw.h
drivers/scsi/qlogicpti_asm.c
```

نحن نعلم أنه جهاز وايرلس USB، ونلاحظ أنه إما جهاز ATM أو SCS ولذا يمكننا بأمان أن نتجاهل الملفات الموجودة في مجلدات atm و scsi.

وذلك يدعنا نتحقق من اسم الملف *drivers/et/wireless/zd1211rw/zd_usb.c*.ويعرض لنا zd_usb.c العبارة 157e في القطعة التالية من الشفرة:

```
static struct usb_device_id usb_ids[] = {
        /* ZD1211 */
        { USB_DEVICE(0x0ace, 0x1211), .driver_info =
DEVICE_ZD1211 },
        { USB_DEVICE(0x07b8, 0x6001), .driver_info =
DEVICE_ZD1211 },
        { USB_DEVICE(0x126f, 0xa006), .driver_info =
DEVICE_ZD1211 },
        { USB_DEVICE(0x6891, 0xa727), .driver_info =
DEVICE_ZD1211 },
        { USB_DEVICE(0x0df6, 0x9071), .driver_info =
DEVICE_ZD1211 },
        { USB_DEVICE(0x157e, 0x300b), .driver_info =
DEVICE_ZD1211 },
        /* ZD1211B */
        { USB_DEVICE(0x0ace, 0x1215), .driver_info =
DEVICE_ZD1211B },
        { USB_DEVICE(0x157e, 0x300d), .driver_info =
DEVICE_ZD1211B },
        {}
};
```

وتشبه مشغلات أجهزة USB مشغلات أجهزة PCI في أنها تخبر النواة بالأجهزة التي تدعمها، كي تقوم النواة بالربط بين المشغل والجهاز.وذلك يحدث عن طريق صيغة المتغير usb_device_id كما يتضح هنا.

وهذه قائمة من أرقام هوية -IDs-مختلفة للبائعين والأجهزة المدعومة بهذه المشغلات :

USB_DEVICE(0x157e, 0x300b), .driver_info = }
,{ DEVICE_ZD1211

هذا السطر يبين أن IDs البائع والمنتج الخاص بنا مدعوم من هذا المشغل .
وبمجرد حصولك على اسم المشغل اللازم للتحكم بهذا الجهاز، عد مرة أخرى خلال *Makefiles* النواة كما تم شرحه من قبل في هذا الفصل لتحديد كيفية تفعيل هذا المشغل لبنائه بشكل سليم.
وفي هذا الملخص توجد الخطوات اللازمة لإيجاد مشغل USB الذي سيتحكم في الجهاز المحدد من نوع usb :

1. أوجد ID البائع والمنتج الخاصة بالجهاز الذي تريد إيجاد مشغل له، باستخدام الأمر *lsusb* بعد توصيل الجهاز بالحاسب، وبعدها قم بنزع الجهاز لترى التغير الحاصل في القائمة.
2. ابحث في شجرة الملف المصدري للنواة عن ID البائع والمنتج الخاصة بجهاز USB. وكل من ID البائع والمنتج يجب وجودها في صيغة التعريف .usb_device_id
3. ابحث في kernel Makefiles عن CONFIG_ rule الذي يقوم ببناء هذا المشغل باستخدام الأمرين *find* و *grep* :

```
$ find -type f -name Makefile | xargs grep DRIVER_NAME
```

4. ابحث في برنامج تهيئة النواة عن قيمة التهيئة، واذهب إلى مكانها في القائمة لتفعيل المشغل ليتم بناؤه.

نظام ملفات الجذر-*Root Filesystem*

Root Filesystem هو ذلك الجزء من نظام الملفات الذي يقلع منه النظام.وهو يحتوي على كل البرامج الأولية التي يبدأ من خلالها إقلاع التوزيعة، ويحتوي عادة على الإعدادات الكاملة للنظام على الحاسب.وباختصار هو مهم جدا ويجب تفعيله وإمكانية عثور النواة عليه وقت الإقلاع، كي تعمل كافة الأمور بشكل سليم.
إذا حدث موت للنواة الجديدة الخاصة بك التي قمت بتهيئتها عند الإقلاع ستحصل على رسالة خطأ شبيهة بما يلي :

```
VFS: Cannot open root device hda2 (03:02)
Please append a correct "root=" boot option
```

```
Kernal panic: VFS: Unable to mount root fs on 03:02
```

وذلك يعني أن root filesystem لم يتم العثور عليه. فإذا لم تكن تستخدم صورة قرص ذاكرة ramdisk image عند الإقلاع ، فمن الموصى به عادة أن تقوم ببناء كل من نظام الملفات الخاص بقسم الروت، ومتحكم القرص الصلب بداخل النواة بدلا من بنائه كـ module.إذا قمت باستخدام ramdisk أثناء الإقلاع يجب عليك حفظ إعدادات بناء هذا الجزء كـ module.



كيف يمكنك أن تقرر ما إذا كنت ستستخدم ramdisk عند الإقلاع ؟

في الفصل الخامس قمنا بالتنويه عن استخدام سكربت لتثبيت التوزيعة يقوم بتثبيت النواة بدلا من القيام بتثبيتها بنفسك.إذا كنت تستخدم سكربت تثبيت التوزيعة فمن المحتمل استخدامك ل ramdisk. فإذا قمت بتثبيتها بنفسك فمن المحتمل أنك لن تفعل.

هذه الأقسام الفرعيةتوضح كيفية ترك النواة تعثر على ال root filesystem أثناء الإقلاع .

نوع نظام الملفات

أولا :يجب تحديد نظام الملفات الذي يستخدمه root partition .ولفعل ذلك انظر إلى ناتج الأمر *mount*:

```
$ mount | grep "/"
/dev/sda2/ on type ext3 (rw,noatime)
```

نحن نهتم بماهية نوع نظام الملفات الذي عرض بعد كلمة type. وفي هذا المثال نوعه ext3.هذا هو نوع نظام الملفات الذي يستخدمه root partition.اذهب إلى برنامج تهيئة النواة وتأكد أن هذا النوع من نظام الملفات مفعل، كما تم شرحه في الفصل الثامن.

متحكم القرص

يتبين من ناتج الأمر *mount* فيما سبق أن الجزء الأول من السطر يوضح ما هو الجهاز الكتلي block device المركب عليه root filesystem .
وفي هذا المثال هو/dev/sda2/.

الآن وقد تم إعداد نظام الملفات بشكل سليم على نواتك، يجب عليك أيضا التأكد من أن الجهاز الكتلي سيعمل كذلك بشكل سليم.

جميع الأجهزة الكتلية تظهر في *sysfs* إما في المسار */sys/block/* أو */class/block* ويعتمد هذا على إصدار النواة التي تستخدمها.

والأجهزة الكتلية تبدو كشجرة-tree، والبارتشنات هي أوراق للجهاز الرئيسي :

```
$ tree -d /sys/block/ | egrep "hd|sd"
|-- hdc
|-- hdd
`-- sda
    |-- sda1
    |-- sda2
    |-- sda3
```

نظرا للمعلومات المعطاة في الأمر *mount* فأنت تحتاج للتأكد من أنه تم تهيئة القسم sda2 بشكل سليم. لأنه بارتشن (أقسام القرص مرقمة بينما الأجهزة الكتلية block devices ليست كذلك)،وكل أقسام القرص sda يجب تهيئتها(وبدون جهاز الكتلة الرئيسي لا يمكن الوصول إلى الأقسام القائمة على هذا الجهاز).

الجهاز الكتلي sda يتم تمثيله كجهاز الشبكة الذي بحثنا عنه مسبقا في هذا الفصل.

يوجد رابط - symlink- لدليل الأجهزة يسمى device يشير للجهاز المنطقي logical device المتحكم في هذا الجهاز الكتلي :

```
$ ls -l /sys/block/sda
  ...
device ->
../../devices/pci0000:00/0000:00:1f.2/host0/target0:0:0/0:0:0
:0
  ...
```

والآن أنت تحتاج لبدء جولة في سلسلة من الأجهزة في *sysfs* لاكتشاف ما هو المشغل الذي يتحكم في ذلك الجهاز :

```
$ ls -l
/sys/devices/pci0000:00/0000:00:1f.2/host0/target0:0:0/0:0:0:
0
  ...
driver -> ../../../../../../bus/scsi/drivers/sd
 ...
```

نرى هنا أن مشغل متحكم أقراص SCSI هو المسئول عن عمل هذا الجهاز .ولذلك فنحن نعلم أننا بحاجة إلى دعم أقراص SCSI في برنامج تهيئة النواة .

واصل العمل في سلسلة المجلدات في *sys* محاولا العثور على مكان المشغل الذي

يتحكم في العتاد :

```
$ ls -l /sys/devices/pci0000:00/0000:00:1f.2/host0/target0:0:0
 ...
```

لا يوجد رابط يسمى driver في هذا المسار، ولذا ارجع خطوة للخلف :

```
$ ls -l /sys/devices/pci0000:00/0000:00:1f.2/host0
```

ومرة أخرى لا يوجد مشغل هنا،واصل البحث وارجع خطوة أخرى للوراء :

```
$ ls -l /sys/devices/pci0000:00/0000:00:1f.2
 ...
driver -> ../../../bus/pci/drivers/ata_piix
 ...
```

ها هو ذا ! ذلك هو متحكم القرص الذي نحتاج التأكد منه في إعداد النواة الخاصة بنا. ولذا من أجل root filesystem نحن نحتاج إلى تفعيل مشغلات ext3 و sd و ata_piix في تهيئة النواة الخاصة بنا حتى يمكننا الإقلاع بنجاح بالنواة على هذا العتاد.

سكربت مساعد :

وكما تم التنويه منذ قليل في مقدمة هذا الفصل ، تتغير الملفات والمجلدات داخل *sysfs* من إصدار للنواة إلى إصدار آخر .وهنا سكربت سهل يقوم بالتحقق من مشغل النواة المطلوب، وتركيب module name لأي device node في النظام .وقد تم تطويره على مسئولية مطوري النواة من أجل *sysfs* وسوف يعمل بنجاح في جميع الإصدارات القادمة من نواة 2.6.

على سبيل المثال ، فهو يقلل من العمل كما في المثال السابق ، عندما كان عليك الحصول على المشغلات المناسبة لكل ال block device الخاصة بالقرص sda :

```
$ get-driver.sh sda
looking at sysfs device: /sys/devices/pci0000:00/0000:00:1f.2/host0/
target0:0:0/0:0:0:0
found driver: sd
found driver: ata_piix
```

يمكن أيضا إيجاد كل المشغلات المناسبة اللازمة لأعمال معقدة مثل محول الأجهزة USB-to-serial :

```
$ get-driver.sh ttyUSB0
looking at sysfs device:
/sys/devices/pci0000:00/0000:00:1d.3/usb4/4-2/4-2.
3/4-2.3:1.0/ttyUSB0
found driver: pl2303 from module: pl2303
found driver: pl2303 from module: pl2303
found driver: usb from module: usbcore
found driver: usb from module: usbcore
```

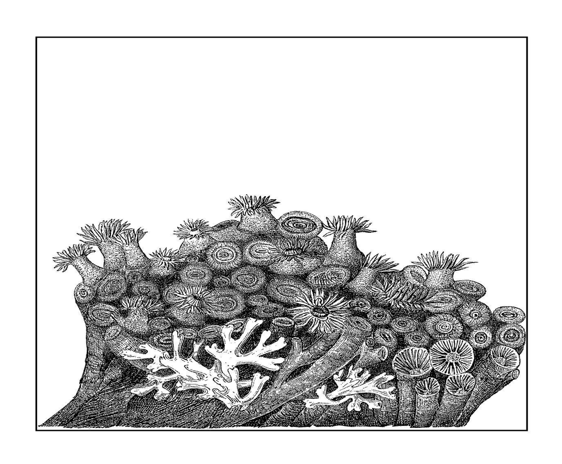
```
found driver: usb from module: usbcore
found driver: uhci_hcd from module: uhci_hcd
```

يمكنك تحميل ملف توضيحي يحتوي على هذا السكربت من الموقع الإلكتروني للكتب المذكور في الافتتاحية بعنوان " كيف تتواصل معنا"

والسكربت كما يلي :

```
#!/bin/sh
#
# Find all modules and drivers for a given class device.
#
if [ $# != "1" ] ; then
    echo
    echo "Script to display the drivers and modules for a
specified sysfs
class device"
    echo "usage: $0 <CLASS_NAME>"
    echo
    echo "example usage:"
     echo "        $0 sda"
     echo "Will show all drivers and modules for the sda
block device."
     echo
     exit 1
fi
DEV=$1
if test -e "$1"; then
     DEVPATH=$1
else
     # find sysfs device directory for device
     DEVPATH=$(find /sys/class -name "$1" | head -1)
     test -z "$DEVPATH" && DEVPATH=$(find /sys/block -name
"$1" | head -1)
     test -z "$DEVPATH" && DEVPATH=$(find /sys/bus -name "$1"
| head -1)
     if ! test -e "$DEVPATH"; then
          echo "no device found"
          exit 1
     fi
fi
echo "looking at sysfs device: $DEVPATH"
if test -L "$DEVPATH"; then
     # resolve class device link to device directory
     DEVPATH=$(readlink -f $DEVPATH)
     echo "resolve link to: $DEVPATH"
fi
if test -d "$DEVPATH"; then
     # resolve old-style "device" link to the parent device
```

```
     PARENT="$DEVPATH";
     while test "$PARENT" != "/"; do
          if test -L "$PARENT/device"; then
               DEVPATH=$(readlink -f $PARENT/device)
               echo "follow 'device' link to parent:
$DEVPATH"
               break
          fi
          PARENT=$(dirname $PARENT)
     done
fi
while test "$DEVPATH" != "/"; do
     DRIVERPATH=
     DRIVER=
     MODULEPATH=
     MODULE=
     if test -e $DEVPATH/driver; then
          DRIVERPATH=$(readlink -f $DEVPATH/driver)
          DRIVER=$(basename $DRIVERPATH)
          echo -n "found driver: $DRIVER"
          if test -e $DRIVERPATH/module; then
               MODULEPATH=$(readlink -f $DRIVERPATH/module)
               MODULE=$(basename $MODULEPATH)
               echo -n " from module: $MODULE"
          fi
          echo
     fi
     DEVPATH=$(dirname $DEVPATH)
done
```



8

وصفات إعداد النواة

الفصول القادمة علمتنا آليات إعادة تهيئة النواة؛ والمحصلة من هذا الفصل هي أين يمكنك الحصول على جميع أنواع التغييرات الأكثر شيوعا والتي يحتاجها الناس ليصنعوا نواتهم الخاصة، مع تعليمات محددة حول كيفية القيام بذلك.

الأقراص:

تدعم نواة لينكس قطاعا عريضا من مختلف أنواع الأقراص.وهذا القسم يوضح كيفية إعداد النواة كي تدعم أغلب أنواع متحكمات الأقراص الأكثر شيوعا.

أجهزة USB :

لاستخدام وسائط تخزين أجهزة USB (الشائع استخدامها كذاكرة الفلاش USB أو الأقراص الصلبة الخارجية من نوع usb) يجب أن يكون جهاز ال USB يعمل بشكل سليم في المقام الأول. وأنا أشير إلى وصفة في هذا الباب تسمى "USB" لكيفية القيام بذلك.

إن جهاز تخزين USB يمكن التعرف عليه باستخدام البرنامج *lsusb* إذا كانت سلسلة الأمر التالي تعطيك النتائج التي تراها، فذلك يعني وجود جهاز تخزين USB على نظامك :

```
$ /usr/sbin/lsusb -v | grep Storage
    bInterfaceClass         8 Mass Storage
```

قم بتفعيله على الوجه التالي:

1- إن جهاز تخزين ال USB هو في الواقع جهاز USB SCSI والذي يتخاطب عبر اتصال USB. وبسبب ذلك يجب تفعيل نظام SCSI :

Device Drivers
  SCSI Device Support
    [*] SCSI Device Support

2- أيضا في نظام scsi يجب تفعيل الخيار (SCSI disk support) من أجل أن يتم وصل (mount) الجهاز بشكل سليم.

Device Drivers
  SCSI Device Support
    [*] SCSI disk support

3- قم بتفعيل الخيار USB Storage support :

Device Drivers
  USB Support
  [M] USB Mass Storage support

هناك عدد محدد من أجهزة تخزين USB مدرجة في قائمة منفصلة في عملية إعداد العناصر، وكأنها لا تتبع المواصفات القياسية لأجهزة USB وتتطلب شفرة خاصة. فإذا كان لديك واحد من هذه الأجهزة، برجاء تفعيل الخيار الذي يدعمها.

أقراص IDE

أقراص IDE أكثر أنواع الأقراص شيوعا في الحاسبات الشخصية.والجهاز الذي يتيح لها العمل على وجه سليم هو متحكم القرص IDE disk controller.وللتحقق من وجود IDE disk controller على نظامك استخدم الأمر *lspci* بالشكل التالي[*]:

```
$ /usr/sbin/lspci | grep IDE
00:1f.1 IDE interface: Intel Corporation 82801EB/ER
(ICH5/ICH5R) IDE
Controller (rev 02)
00:1f.2 IDE interface: Intel Corporation 82801EB (ICH5) SATA
Controller (rev02)
```

لاحظ أن النتيجة عندك قد تكون غير مطابقة بالضبط للنتيجة السابقة، ولكن المهم أن هذا الأمر يريك بعض متحكمات ال IDE (وهو أول جهاز في المثال السابق) فإذا

(*) كل التوزيعات تضع برنامج lspci في المسار /usr/sbin/ .ولكن بعض بعضها يضعه في مكان مختلف.وللعثور على مكان وضع البرنامج اكتب:

```
$ which lspci
/usr/sbin/lspci
```

إذا كنت تستخدم توزيعة تضع البرنامج في مكان ما ، فنرجو استخدام هذا المسار كلما تحدثنا عن الأمر *lspci*.

وجدت متحكمات SATA فحسب، فمن فضلك انظر في القسم التالي ("Serial
SATA "ATA) والآن قم بتطبيق الخطوات التالية :

1-قم بتفعيل PCI support للنواة :

Bus options (PCI, PCMCIA, EISA, MCA, ISA)
[*] PCI Support

2- قم بتفعيل نظام IDE الفرعي و IDE support

Device Drivers
[*] ATA/ATAPI/MFM/RLL support
[*] Enhanced IDE/MFM/RLL disk/cdrom/tape/floppy support

3- وفي نظام ATA يجب تفعيل النوع المحدد من IDE controller كي يعمل بشكل سليم .ولكي تزود بنسخة احتياطية جيدة في حالة اختيارك للنوع الخطأ، اختر الخيار "generic" لمتحكم ال IDE

Device Drivers
ATA/ATAPI/MFM/RLL support
[*] generic/default IDE chipset support

4. قم بتفعيل انواع متحكمات أجهزة PCI IDE المختلفة :

Device Drivers
ATA/ATAPI/MFM/RLL support
[*] PCI IDE chipset support

وذلك يفتح لك قائمة فرعية طويلة من مختلف أنواع IDE controller. اختر منها ما يناسبك اعتمادا على اسم الجهاز الذي وجدته عند استخدام الأمر *lspci*.

أجهزة الساتا

sata هو نوع من متحكمات الأقراص التي ورثت متحكم أقراص IDE على النظام.
ولتحديد ما إذا كان لديك قرص ساتا على نظامك، اكتب الأمر التالي :

```
$ /usr/sbin/lspci | grep SATA
00:1f.2 IDE interface: Intel Corporation 82801EB (ICH5) SATA
Controller (rev 02)
```

لاحظ أن النتيجة عندك ربما تكون غير مطابقة لهذا المثال، ولكن ما يهم هنا هو معرفة الأمر الذي يعرض لك بعض أجهزة sata.

تستخدم أقراص ساتا مكتبة للنواة تدعى *libata* والتي تتعامل مع أغلب أنواع الوظائف لهذه الأقراص.هذه المكتبة تستخدم طبقة SCSI للتخاطب مع طبقة block layer، حيث أن هناك أنواع متعددة من خيارات النواة تحتاج للتفعيل كي تعمل أقراص ساتا بشكل سليم .

1- قم بتفعيل PCI support للنواة

Bus options (PCI, PCMCIA, EISA, MCA, ISA)
  [*] PCI Support

2. قم بتفعيل SCSI subsystem

Device Drivers
  SCSI Device Support
    [*] SCSI Device Support

3. كذلك في نظام SCSI فإن SCSI disk تدعم خيارا لابد من تفعيله ليتم وصل "mount" الجهاز بشكل صحيح.

Device Drivers
  SCSI Device Support
    [*] SCSI disk support

4.ستجد خيارات sata تحت القسم "SCSI low-level drivers":

Device Drivers
  SCSI Device Support
    SCSI low-level drivers
      [*] Serial ATA (SATA) support

5. في هذا القسم بتفعيل نوع متحكم الساتا المخصص الذي لديك.

انظر في ناتج الأمر السابق *lspci*، لتحصل على قائمة من أنواع متحكمات الساتا sata controllers التي توجد على نظامك.

على سبيل المثال معظم اللوحات الأم التابعة لإنتل تحتاج إلى مشغل PIIX/ICH SATA (كما هو موضح في المثال السابق):

Device Drivers
  SCSI Device Support
    SCSI low-level drivers
      [*] Serial ATA (SATA) support
      [*]   Intel PIIX/ICH SATA support

حرق الأقراص المضغوطة :

عملية حرق قرص مضغوط على لينكس في غاية البساطة، فإذا كانت نواتك تدعم القراءة من القرص المضغوط، فإنه يمكنها أيضا أن تدعم حرق القرص المضغوط.
وهناك طريقتان لدعم القرص المضغوط في لينكس، أحداهما لسواقات IDE والأخرى لسواقات SATA .

IDE CD-ROM drives

يتم التحكم في سواقات IDE CD-ROM من خلال نفس المتحكم IDE controller كما هو الحال مع الأقراص الصلبة . كن متأكدا من أن IDE controller مدعوم كما تم شرحه في الجزء الخاص بأقراص IDE، فإذا كان مدعوما بشكل صحيح فإنه عليك اختيار إعداد عنصر واحد وهو :

Device Drivers
    [*] ATA/ATAPI/MFM/RLL support
    [*] Enhanced IDE/MFM/RLL disk/cdrom/tape/floppy support
    [M]   Include IDE/ATAPI CDROM support

SCSI and SATA CD-ROM drives

يتم التحكم في سواقات الأقراص المضغوطة من نوع سكازي وساتا من خلال نفس متحكم أقراص القرص الصلب الرئيسي لديك . كن متأكدا من دعم متحكم ساتا وسكازي لديك بشكل سليم .
ولأقراص ساتا انظر في القسم السابق :(SATA) (Serial ATA - SATA).
ولدعم سواقات SATA أو SCSI CD-ROM يجب تفعيل SCSI CD-ROM driver

Device Drivers
    SCSI Device Support
        [*] SCSI CDROM support

Devices

إن نظام لينكس يدعم نطاقا واسعا من الأنواع المختلفة للأجهزة (أكثر مما يقوم به أي نظام تشغيل آخر)
وهذا القسم يوضح كيفية تفعيل بعض هذه الأنواع الأكثر شيوعا .

USB

يدعم لينكس العديد من الأنواع المختلفة من أجهزة USB . ولتمكين دعم USB

عليك أولا أن تفعل USB controller والتي تقود اتصال USB على الجهاز.
وللتحقق من وجود USB controller على جهازك ومعرفة نوعه اكتب الأمر التالي :

```
$ /usr/sbin/lspci | grep USB
```

والنتيجة كالتالي:

```
00:1d.0 USB Controller: Intel Corporation 82801EB/ER
(ICH5/ICH5R) USB UHCI
Controller #1 (rev 02)
00:1d.1 USB Controller: Intel Corporation 82801EB/ER
(ICH5/ICH5R) USB UHCI
Controller #2 (rev 02)
00:1d.2 USB Controller: Intel Corporation 82801EB/ER
(ICH5/ICH5R) USB UHCI
Controller #3 (rev 02)
00:1d.3 USB Controller: Intel Corporation 82801EB/ER
(ICH5/ICH5R) USB UHCI
Controller #4 (rev 02)
00:1d.7 USB Controller: Intel Corporation 82801EB/ER
(ICH5/ICH5R) USB2 EHCI
Controller (rev 02)
```

لاحظ أن النتيجة التي تحصل عليها ربما تكون غير متطابقة مع هذا المثال، ولكن المهم هو معرفة الأمر الذي يعرض لك بعض USB controllers.

1- قم بتفعيل PCI support للنواة

```
Bus options (PCI, PCMCIA, EISA, MCA, ISA)
   [*] PCI Support
```

2- قم بتفعيل USB support للنواة

```
Device Drivers
   USB Support
      [M] Support for Host-side USB
```

3- قم بتفعيل USB Host controllers على جهازك(ومن الأمان أن تقوم بتفعيلها كلها إن كنت لا تعرف أيا منها هو الموجود لديك):

```
Device Drivers
   USB Support
   ---   USB Host Controller Drivers
```

[M] EHCI HCD (USB 2.0) support
[M] OHCI HCD support
[M] UHCI HCD (most Intel and VIA) support

4- الجهاز المنفرد ل USB يحتاج أيضا لتفعيل هذه المشغلات. ويوجد عدد رئيسي وكبير منها تحت القسم الرئيسي لمشغل USB :

```
Device Drivers
   USB Support
```

لكن بعض الأجهزة مثل USB video و DVB و sound توجد في كل قائمة خاصة بمتحكمات هذه الأجهزة، فعلى سبيل المثال مشغل جهاز USB sound يمكن وجوده تحت القائمة Sound:

```
Device drivers
   Sound
      [*] Sound card support
         [*] Advanced Linux Sound Architecture
            USB Devices
               [M] USB Audio/MIDI driver
```

إذا كنت تريد إدراج جهاز تخزين من نوع usb (usb flash)، انظر الآن في القسم المسمى USB Storage في بداية هذا الفصل .

IEEE 1394 (FireWire)

(IEEE 1394 (FireWire يعرف باسم شائع وهو FireWire ، وذلك الاسم الذي نشرته شركة أبل للحاسبات. IEEE 1394 هو أحد النواقل عالية السرعة التي توصل بها الأجهزة الخارجية "external" مثلما تفعل أجهزة USB.
وللتحقق من وجود متحكم ل FireWire على جهازك، ومعرفة نوعه اكتب الأمر التالي :

```
$ /usr/sbin/lspci | grep FireWire
06:0c.0 FireWire (IEEE 1394): Texas Instruments TSB43AB22/A
IEEE-1394a-2000
Controller (PHY/Link)
06:0d.2 FireWire (IEEE 1394): Creative Labs SB Audigy
FireWire Port (rev 04)
```

لاحظ أن الناتج الذي تحصل عليه ربما لا يتطابق مع هذا المثال ولكن المهم هو مرفة الأمر الذي يعرض لك بعض متحكمات FireWire .

1- قم بتفعيل الخيار PCI support للنواة :

Bus options (PCI, PCMCIA, EISA, MCA, ISA)
    [*] PCI Support

2- قم بتفعيل الخيار IEEE 1394 support للنواة:

Device Drivers
    IEEE 1394 (FireWire) support
        [*] IEEE 1394 (FireWire) support

3- قم بتفعيل النوع المحدد لمتحكم جهاز FireWire المضيف الموجود لديك :

Device Drivers
    IEEE 1394 (FireWire) support
        [*] IEEE 1394 (FireWire) support
        ---   Device Drivers
        [M]   Texas Instruments PCILynx support
        [M]   OHCI-1394 support

4- وأخيرا قم بتفعيل الأنواع المحددة لأجهزة FireWire الموجودة لديك :

Device Drivers
    IEEE 1394 (FireWire) support
        [*] IEEE 1394 (FireWire) support
        ---   Protocol Drivers
        [M]   OHCI-1394 Video support
        [M]   SBP-2 support (Harddisks etc.)
        [ ]   Enable Phys DMA support for SBP2 (Debug)
        [M]   Ethernet over 1394
        [M]   OHCI-DV I/O support
        [M]   Raw IEEE1394 I/O support

:PCI Hotplug

أصبحت أنظمة PCI Hotplug أكثر الأجهزة شعبية والتي تسخدم في ExpressCard ومحطات عمل الأجهزة المحمولة .

للتحقق إذا كان في جهازك متحكم ExpressCard على جهازك، ابحث في العتاد لترى إذا كان ممكنا لبطاقة ExpressCard أن توصل بها.

1- قم بتفعيل PCI support للنواة :

Bus options (PCI, PCMCIA, EISA, MCA, ISA)
[*] PCI Support

2- قم بتفعيل PCI hotplug support للنواة :

Bus options (PCI, PCMCIA, EISA, MCA, ISA)
[*] PCI Support
PCI Hotplug Support
[M] Support for PCI Hotplug (EXPERIMENTAL)

3- هناك نطاق واسع ومختلفة من أنواع أجهزة المتحكمات PCI hotplug لدعم أغلب أجهزة المحمول وبطاقات ExpressCard،
قم بتفعيل متحكم ACPI :

Bus options (PCI, PCMCIA, EISA, MCA, ISA)
[*] PCI Support
PCI Hotplug Support
[M] Support for PCI Hotplug (EXPERIMENTAL)
[M] ACPI PCI Hotplug driver

4- كذلك قم بتفعيل متحكم PCI Express :

Bus options (PCI, PCMCIA, EISA, MCA, ISA)
[*] PCI Support
[*] PCI Express Support
[M] PCI Express Hotplug driver

PCMCIA/CardBus

دعم أجهزة PCMCIA و CardBus موجود في جميع الحاسبات المحمولة المصنعة. الحواسب المحمولة الحديث - أيا كانت- تتحول إلى ExpressCard،
(انظر إلى الوصفة الخاصة بـ PCI Hotplug في القسم السابق "PCI Hotplug”)

للتحقق إذا ما كان على جهازك متحكم PCMCIA ، انظر في قائمة العتاد، وإذا ما كان ممكنا لبطاقة PCMCIA أن تتصل به .

1- قم بتفعيل PCI suppor للنواة :

Bus options (PCI, PCMCIA, EISA, MCA, ISA)
[*] PCI Support

2- قم بتفعيل PCCARD support للنواة :

Bus options (PCI, PCMCIA, EISA, MCA, ISA)
PCCARD (PCMCIA/CardBus) support
[M] PCCard (PCMCIA/CardBus) support

3- قم بتفعيل كل من PCMCIA و CardBus support لتغطي أوسع نطاق من الأجهزة :

Bus options (PCI, PCMCIA, EISA, MCA, ISA)
PCCARD (PCMCIA/CardBus) support
[M] PCCard (PCMCIA/CardBus) support
[M] 16-bit PCMCIA support
[*] 32-bit CardBus support

قم بتفعيل نوع جسر البطاقة card bridge لحاسبك المحمول .أغلب الأنواع الشائعة بها متحكم "yenta-like":

Bus options (PCI, PCMCIA, EISA, MCA, ISA)
PCCARD (PCMCIA/CardBus) support
[M] PCCard (PCMCIA/CardBus) support
[M] CardBus yenta-compatible bridge support
[ ] Cirrus PD6729 compatible bridge support
[ ] i82092 compatible bridge support
[ ] i82365 compatible bridge support
[ ] Databook TCIC host bridge support

Sound (ALSA)

المعمارية المتقدمة للصوت في لينكس(Advanced Linux Sound Architecture (ALSA هي النظام الحالي للصوت في نواة لينكس.

وقد تم حذف نظام الصوت الأقدم (Oss).وأغلب المشغلات القديمة تم حذفها من شجرة مصدر نواة لينكس.

للتحقق من وجود متحكم الصوت على جهازك ومعرفة نوعه اكتب الأمر التالي :

```
$ /usr/sbin/lspci | grep -i audio
00:1f.5 Multimedia audio controller: Intel Corporation
82801EB/ER (ICH5/
ICH5R) AC'97 Audio Controller (rev 02)
06:0d.0 Multimedia audio controller: Creative Labs SB Audigy
(rev 04)
```

لا حظ أن الناتج الذي تحصل عليه ربما يكون غير مطابق لهذا المثال ولكن المهم هو

الأمر الذي يعرض لك بعض متحكمات الصوت .

1- قم بتفعيل دعم للصوت الأساسي :

Device Drivers
    Sound
        [M] Sound Card Support

2- قم بتفعيل ALSA

Device Drivers
    Sound
        [M] Sound Card Support
            [M] Advanced Linux Sound Architecture

3- يوجد عدد من الخيارات الأساسية ل ALSA مثل بروتوكول الصوت OSS القديم . إذا كان لديك بعض التطبيقات القديمة يجب عليك تفعيل الخيارات المتعلقة بها :

Device Drivers
    Sound
        [M] Sound Card Support
            [M] Advanced Linux Sound Architecture
            [M] OSS Mixer API
            [M] OSS PCM (digital audio) API
            [ ] OSS PCM (digital audio) API - Include plugin system

4- قم بتفعيل النوع المحدد من جهاز الصوت لديك ،كروت صوت PCI توجد تحت القائمة الفرعية PCI:

Device Drivers
    Sound
        [M] Sound Card Support
            [M] Advanced Linux Sound Architecture
                PCI Devices

CPU

إذا كنت تريد أن تعمل نواة لينكس بأقصى سرعة ممكنة من خلال المعالج والعتاد

المحدد الموجود لديك، فهناك عدة خيارات قليلة يمكنك وضعها لتحصل على أقصى نقطة في أداء قطع العتاد لديك. هذا القسم سوف يريك بعض الخيارات لمعالجات معينة والتي يمكنك جعلها متناغمة مع معالجك .

Processor Types

هناك قطاع واسع من خيارات المعالجات متاحة للتغيير فيما بينها في نواة لينكس . والهدف الأهم لنا هو تحديد نوع وحدة المعالجة المركزية التي تستخدمه لنواتك بالضبط . لتحديد نوع المعالج الذي تستخدمه اكتب الأمر التالي :

```
$ cat /proc/cpuinfo | grep "model name"
model name      : Intel(R) Xeon(TM) CPU 3.20GHz
```

لاحظ أن نتيجتك قد لا تكون مطبقة لهذا المثال، ولكن المهم هو معرفة الأمر الذي يعرض لك نوع المعالج الموجود على نظامك .

1- قم باختيار نوع معمارية المعالج :

```
Processor type and features
    Subarchitecture Type
        (X) PC-compatible
        ( ) AMD Elan
        ( ) Voyager (NCR)
        ( ) NUMAQ (IBM/Sequent)
        ( ) Summit/EXA (IBM x440)
        ( ) Support for other sub-arch SMP systems with more than 8
CPUs
        ( ) SGI 320/540 (Visual Workstation)
        ( ) Generic architecture (Summit, bigsmp, ES7000, default)
        ( ) Support for Unisys ES7000 IA32 series
```

إذا كان جهازك يحتوي على معالج مختلف عن هذه الأنواع الموجودة في القائمة السابقة يجب عليك اختيار أي شيء آخر غير خيار PC-compatible.

على أية حال إذا كنت ترغب في إنشاء نواة وحيدة يمكنك تشغيل كل أنواع الأجهزة المعروضة عليها قم بتحديد الخيار Generic architecture .

بعض الخيارات أعلاه ربما لا تكون موجودة إذا لم تقم أيضا بتحديد خيار دعم تعدد المعالجات Symmetric multiprocessing

2- حدد عائلة المعالج. ويجب تحديد خيار PC-compatible من الخيار السابق لهذه القائمة الفرعية لكي يتم عرضها :

Processor type and features
    Processor family
        ( ) 386
        ( ) 486
        ( ) 586/K5/5x86/6x86/6x86MX
        ( ) Pentium-Classic
        ( ) Pentium-MMX
        ( ) Pentium-Pro
        ( ) Pentium-II/Celeron(pre-Coppermine)
        ( ) Pentium-III/Celeron(Coppermine)/Pentium-III Xeon
        ( ) Pentium M
        (X) Pentium-4/Celeron(P4-based)/Pentium-4 M/Xeon
        ( ) K6/K6-II/K6-III
        ( ) Athlon/Duron/K7
        ( ) Opteron/Athlon64/Hammer/K8
        ( ) Crusoe
        ( ) Efficeon
        ( ) Winchip-C6
        ( ) Winchip-2
        ( ) Winchip-2A/Winchip-3
        ( ) GeodeGX1
        ( ) Geode GX/LX
        ( ) CyrixIII/VIA-C3
        ( ) VIA C3-2 (Nehemiah)
        ( ) Generic x86 support

لمزيد من التفاصيل الخاصة بإعداد هذا العنصر ارجع إلى الخانةالخاصة ب M386 في الفصل 11 لمعرفة الموصفات الكاملة لكيفية انتقاء النوع المناسب للمعالج اعتمادا على المعالج الذي لديك، وما هي مجموعة المعالج التي تريد للنواة أن تعمل عليها .

SMP

وإذا كان النظام الخاص بك يحتوي على أكثر من معالج ، أو ثنائية أو Hyperthreaded ، يجب عليك تحديد خيار تعدد المعالج multiprocessor

لنواة لينكس من أجل الاستفادة من المعالجات الإضافية. وإن لم تفعل ذلك ، ستخسر الاستفادة من المعالجات الأخرى بعدم استخدامك لها جميعا .

قم بتفعيل تعدد المعالجة multiprocessing :

Processor type and features

[*] Symmetric multi-processing support

Preemption

الأجهزة التي تعمل كخوادم تختلف في متطلبات عبء العمل عن تلك التي تعمل كأجهزة سطح مكتب لتشغيل تطبيقات الصوت والصورة. تسمح النواة لأنماط مختلفة من حق الشفعة أو المشاركة(preemption) في سبيل التعامل مع تلك الأعباء المختلفة من العمل .

preemption هو قدرة النواة على مقاطعة عمل لها، في الوقت الذي تقوم فيه بعمل شيء آخر، من أجل العمل على شيء ما مع أولوية أعلى، مثل تحديث برنامج صوت أو فيديو .

وللتحول إلى preemption model مختلف استخدم هذه القائمة :

Processor type and features

Preemption Model

(X) No Forced Preemption (Server)

( ) Voluntary Kernel Preemption (Desktop)

( ) Preemptible Kernel (Low-Latency Desktop)

إذا أردت أن تجعل النواة أكثر استجابة مع المهام الأعلى أولوية أكثر مما يمدنا به خيار preemption العام، يمكنك أيضا السماح بالمقاطعات لأحد الأقفال الداخلية الرئيسية في النواة Kernel Lock :

Processor type and features

[*] Preempt The Big Kernel Lock

هذا الخيار قابل لعدم التحديد فقط في حالة تحديدك لخيار Preemptible Kernel أوخيار Symmetric multi-processing .

Suspend

تملك نواة لينكس القدرة على عمل توقف Suspend-أو تعليق- لنفسها وتسمح لك بإغلاق الجهاز، وبعد ذلك عند تشغيل الجهاز تعود بالضبط إلى ما كان عليه النظام

قبل توقف الجهاز .هذه الوظيفة مفيدة جدا للحاسبات المحمولة التي تستخدم لينكس
.

قم بتفعيل ذلك عن طريق تحديد الخيار :

Power management options (ACPI, APM)
[*] Software Suspend

تحتاج النواة لمعرفة أين تحفظ صورة توقف النواة "suspended kernel image" ، ومعرفة من أين تستعيدها فيما بعد. هذا المكان يكون عادة هو قسم السواب على القرص الصلب ، ولتحديد القسم الذي يجب إعداده :

Power management options (ACPI, APM)
(/dev/hda3) Default resume partition

كن متأكدا أنك اخترت القسم المناسب لحفظ عملية تعليق الجهاز، ولا تستخدم القسم المستخدم من قبل النظام لحفظ البيانات. اسم القسم المناسب يمكنك العثور عليه باستخدام الأمر التالي :

```
$ /sbin/swapon -s | grep dev | cut -f 1 -d ' '
/dev/hda3
```

استخدم ناتج الأمر السابق كخيار في إعدادات هذه النواة، في سطر إقلاع النواة عند تحديد المكان الذي يجب على النواة أن تسترجع منه . ولكي يعود النظام للعمل بشكل سليم وبعد عمل تعليق للجهاز عليك تمرير هذه القيمة:

resume=/dev/swappartition[1] إلى سطر أوامر النواة، كي تجعلها تستخدم الـ image الصحيح. إذا لم تكن ترغب في عمل استعادة لـ suspended image استخدم القيمة noresume في سطر أوامر إقلاع النواة.

CPU Frequency Scaling

أغلب المعالجات الحديثة يمكنها إبطاء التردد الداخلي للمعالج لتوفير الطاقة والحفاظ على عمر البطارية . يدعم لينكس هذه الإمكانية، ويقدم العديد من متحكمات-governors- الطاقة المختلفة تنفذ استدلالات مختلفة من أجل تحديد كيفية تغيير سرعة المعالج اعتمادا على عبء النظام ومتغيرات أخرى.

1- قم بتفعيل الوظيفة الأساسية frequency scaling :

Power management options (ACPI, APM)
[*] CPU Frequency scaling

2- اختر نوعا مختلفا من governors frequency والذي ترغب في استخدامه:

---

(1) يتم كتابة رقم البارتشن المناسب لك بدلا من كلمة swappartition

Power management options (ACPI, APM)
    [*] CPU Frequency scaling
    [*]   'performance' governor
    [*]   'powersave' governor
    [*]   'userspace' governor for userspace frequency scaling
    [*]   'ondemand' cpufreq policy governor
    [*]   'conservative' cpufreq governor

لمزيد من المعلومات عما تقوم به governors المختلفة انظر الخانة الخاصة بـ CPU_FREQ في الفصل الحادي عشر.

3- اختر ال governor الافتراضي الذي ترغب في تشغيله عند إقلاع الجهاز.

Power management options (ACPI, APM)
    [*] CPU Frequency scaling
        Default CPUFreq governor (performance)

4- اختر النوع المحدد للمعالج الموجود على جهازك.ولمزيد من التفاصيل عن كيفية اختيار نوع المعالج للجهاز انظر إلى القسم السابق الخاص بـ "Processor Types”

Power management options (ACPI, APM)
    [*] CPU Frequency scaling
    ---   CPUFreq processor drivers
    [ ]   ACPI Processor P-States driver
    [ ]   AMD Mobile K6-2/K6-3 PowerNow!
    [ ]   AMD Mobile Athlon/Duron PowerNow!
    [ ]   AMD Opteron/Athlon64 PowerNow!
    [ ]   Cyrix MediaGX/NatSemi Geode Suspend Modulation
    [*]   Intel Enhanced SpeedStep
    [*]   Use ACPI tables to decode valid frequency/voltage pairs
    [*]   Built-in tables for Banias CPUs
    [ ]   Intel Speedstep on ICH-M chipsets (ioport interface)
    [ ]   Intel SpeedStep on 440BX/ZX/MX chipsets (SMI interface)
    [ ]   Intel Pentium 4 clock modulation
    [ ]   nVidia nForce2 FSB changing
    [ ]   Transmeta LongRun

Different Memory Models

تستطيع نواة لينكس بنظام إنتل 32-bit التعامل مع 64 GB من الذاكرة، ولكن

مساحة العناوين في معالجات 32-bit لا تتعدى 4 GB .ولكي يتم الالتفاف حول هذه المحددات يمكن لنواة لينكس رسم خريطة للذاكرة المضافة داخل مساحة أخرى ومن ثم التحول إليها عندما تحتاج إليها المهام الأخرى. ولكن إذا كان على جهازك مساحة صغيرة من الذاكرة فإنه من اليسير على لينكس ألا يعاني من التعامل مع المساحات الأكبر، لذلك فمن المفيد إخبار النواة بمقدار الذاكرة التي تريد دعمها. ولمزيد من التفاصيل حول مناقشة هذا الخيار ، يرجى الاطلاع على الخانة الخاصة بـ HIGHMEM في الفصل 11.

تدعم نواة لينكس ثلاثة أنماط من الذاكرة لمعالجات إنتل ذات 32-bit اعتمادا على الذاكرة المتاحة :

- أقل من واحد جيجا من الذاكرة الفيزيائية.
- ما بين 1 – 4 جيجا من الذاكرة الفيزيائية.
- أكثر من 4 جيجا من الذاكرة الفيزيائية.

لاختيار مقدار الذاكرة :

Processor type and features
    High Memory Support
        (X) off
        ( ) 4GB
        ( ) 64GB

ACPI

في أغلب الأنظمة القائمة على إنتل تعتبر ACPI لازمة لعمل الجهاز بشكل صحيح .
ACPI [1] معيار قياسي يتيح لنظام الدخل والخرج الأساسي BIOS الخاص بالحاسب للعمل مع نظام التشغيل على تشغيل العتاد بشكل غير مباشر، أملا في معالجة مجموعة واسعة من الأجهزة مع شفرة قليلة نسبيا خاصة بكل نظام تشغيل. كما يوفر ACPI بسهولة المساعدة في خاصية تعليق-suspend، واستئناف-resume عمل الجهاز، والتحكم في سرعة المعالج والمراوح. إذا كان لديك حاسب محمول، فمن الموصى به أن تقوم بتمكين هذا الخيار.

---

(1) ACPI : (Advanced Configuration and Power Interface)(واجهة التطبيق المتقدمة لإدارة الطاقة) هو معيار قياسي مفتوح تم إطلاقه لأول مرة في ديسمبر 1996 وقام على تطويره عدة شركات (Toshiba, Phoenix, Microsoft, Intel, HP, و Dell).. وكانت في الأصل موجهة للحاسبات المحمولة ولكن تم نقلها إلى الحاسبات المكتبية والخوادم ومحطات العمل ،ويمكن من خلالها توفير طاقة الأجهزة الخاملة داخل الحاسوب والعمل بخاصية sleep و hibernate ويمكن عمل إيقاظ للجهاز من خلال ضغطة الفارة أو لوحة المفاتيح حسب إعدادات ال BIOS على جهازك

لتمكين خيار ACPI:

Power management options (ACPI, APM)
    ACPI (Advanced Configuration and Power Interface) Support
        [*] ACPI Support

هناك مساحة واسعة من مشغلات ACPI والتي تتحكم في مختلف أنوواع أجهزة ACPI.يجب عليك تمكين الجهاز المحدد الذي يوجد في جهازك:

Power management options (ACPI, APM)
    ACPI (Advanced Configuration and Power Interface) Support
        [*] ACPI Support
        [*]   AC Adapter
        [*]   Battery
        [*]   Button
        [*]   Video
        [*]   Generic Hotkey (EXPERIMENTAL)
        [*]   Fan
        [*]   Processor
        [*]   Thermal Zone
        [ ]   ASUS/Medion Laptop Extras
        [ ]   IBM ThinkPad Laptop Extras
        [ ]   Toshiba Laptop Extras

: Networking

تعتبر الشبكات من الأمور المطلوبة اليوم في الأعم الأغلب من الأجهزة، يدعم لينكس جميع أنواع الشبكات المتاحة. وهنا سوف أعرض عليك فقط قليلا من الأنواع الموجودة الآن لجميع خيارات التشبيك، بالإضافة إلى المشغلات المختلفة، ويجب تمكين خيار إعداد Networking support الرئيسي

Networking
    [*] Networking support

يجب كذلك تحديد خيار The TCP/IP كي تستطيع الأجهزة التخاطب مع بعضها البعض على شبكة الإنترنت

Networking
        [*] Networking support
                Networking options
                    [*] TCP/IP networking

Netfilter

يعتبر Netfilter جزءا من نواة لينكس يعمل كإطار عمل لترشيح ومعالجة جميع الرزم على الشبكة، والتي تمر عبر الحاسب . ومن الشائع استخدامها إذا كنت ترغب في تفعيل الجدار الناري على جهاز ما لحمايته من نظم مختلفة على شبكة الإنترنت ، أو استخدام الجهاز كوكيل بروكسي لأجهزة أخرى على الشبكة. لمزيد من التفاصيل حول ما يصلح له Netfilter ، يرجى الاطلاع على الخانة الخاصة ب NETFILTER في الفصل 11 .

1- لتفعيل خيار NETFILTER الرئيسي :

```
Networking
   [*] Networking support
         Networking options
            [*] Network packet filtering (replaces ipchains)
```

2- من الموصى به أن تفعل Netfilter netlink interface و Xtables support عند استخدام netlink :

```
Networking
   [*] Networking support
         Networking options
            [*] Network packet filtering (replaces ipchains)
                  Core Netfilter Configuration
                     [*] Netfilter netlink interface
                     [*] Netfilter Xtables support (required for ip_tables)
```

3- البروتوكولات المختلفة التي ترغب في فلترتها يجب اختيارها أيضا :

```
Networking
   [*] Networking support
         Networking options
            [*] Network packet filtering (replaces ipchains)
                  IP: Netfilter Configuration
                     [M] Connection tracking (required for masq/NAT)
                     [ ]   Connection tracking flow accounting
                     [ ]   Connection mark tracking support
                     [ ]   Connection tracking events (EXPERIMENTAL)
                     [ ]   SCTP protocol connection tracking support
                           (EXPERIMENTAL)
                     [M]   FTP protocol support
```

[ ] IRC protocol support
[ ] NetBIOS name service protocol support (EXPERIMENTAL)
[M] TFTP protocol support
[ ] Amanda backup protocol support
[ ] PPTP protocol support
[ ] H.323 protocol support (EXPERIMENTAL)

Network Drivers

تدعم نواة لينكس منظومة واسعة من أجهزة الشبكة. الأعم الأغلب منها هو أجهزة الشبكة الخاصة ب PCI التي يمكن توصيلها مع كابل الإيثرنت .
للتحقق من وجود بطاقة شبكة من نوع PCI على نظامك، وما هو نوعها اكتب الأمر التالي :

$ /usr/sbin/lspci | grep Ethernet

```
03:0c.0 Ethernet controller: D-Link System Inc RTL8139
Ethernet (rev 10)
03:0e.0 Ethernet controller: Intel Corporation 82545GM
Gigabit Ethernet
Controller (rev 04)
```

لاحظ أن الناتج عنك قد يكون غير متطابق مع هذا المثال، لكن المهم هو معرفة الأمر الذي يعرض لك بعضا من أجهزة PCI Ethernet .

1- قم بتفعيل خيار PCI support للنواة :

Bus options (PCI, PCMCIA, EISA, MCA, ISA)
[*] PCI Support

2- قم بتفعيل network device support الرئيسي :

Device Drivers
Network device support
[*] Network device support

3- بعد ذلك تأتي المهمة المبهجة، وهي العثور على المشغلات المخصصة لعتادك.
الأغلب الأعم من أجهزة إيثرنت الحديثة تجدها في قسم gigabit (1000 Mbi) في تحديد ال Driver :

Device Drivers
Network device support

[*] Network device support
Ethernet (1000 Mbit)
بعض أجهزة إيثرنت القديمة سوف تجدها في قسم (10- 100mb-or) :
Device Drivers
Network device support
[*] Network device support
Ethernet (10 or 100Mbit)
ابحث خلال هذه الأقسام لتعثر على المشغل المناسب لأجهزتك المحددة .

IrDA

IrDA هو بروتوكول الأشعة تحت الحمراء الذي يستخدمه عدد من أجهزة الكمبيوتر المحمولة وأجهزة المساعد الرقمي الشخصي للاتصال عبر مسافات قصيرة جدا. وهي منتشرة على الأجهزة القديمة، والأجهزة الأحدث تستخدم البلوتوث في الاتصال بدلا من ذلك. انظر في القسم الأتي من هذا الباب configuring Bluetooth .

1- IrDA هو بروتوكول شبكي، لذلك يمكن أن يوجد تحت قائمة القسم الرئيسي لل networking :

Networking
[*] Networking support
[*] IrDA (infrared) subsystem support

2- يوجد عدد من بروتوكولات IrDA المختلفة يمكن اختيارها، ويعتمد ذلك على نوع الجهاز الذي تريده أن يتصل بالبرنامج ويستخدمه البرنامج في الاتصال :

Networking
[*] Networking support
--- IrDA (infrared) subsystem support
--- IrDA protocols
[*] IrLAN protocol (NEW)
[*] IrCOMM protocol (NEW)
[*] Ultra (connectionless) protocol (NEW)

3- هناك قطاع كبير من مختلف أنواع أجهزة IrDA ، بعضها serial وبعضها PCI والآخر يعتمد على USB. لاختيار نوع محدد من جهاز IrDA الموجود لديك، اختره تحت القائمة الفرعية لمشغل IrDA

Networking
[*] Networking support

--- IrDA (infrared) subsystem support
Infrared-port device drivers
--- SIR device drivers
[ ] IrTTY (uses Linux serial driver)
--- Dongle support
--- Old SIR device drivers
--- Old Serial dongle support
--- FIR device drivers
[ ] IrDA USB dongles
[ ] SigmaTel STIr4200 bridge (EXPERIMENTAL)
[ ] NSC PC87108/PC87338
[ ] Winbond W83977AF (IR)
[ ] Toshiba Type-O IR Port
[ ] SMSC IrCC (EXPERIMENTAL)
[ ] ALi M5123 FIR (EXPERIMENTAL)
[ ] VLSI 82C147 SIR/MIR/FIR (EXPERIMENTAL)
[ ] VIA VT8231/VT1211 SIR/MIR/FIR

Bluetooth

Bluetooth هي تكنولوجيا الوايرلس التي تم اختراعها لتحل محل IrDA للتخاطب فيما بين الأجهزة عبر المسافات الصغيرة جدا. وهي تقنية وايرلس قصير المدى التي تم تصميمها لتحل محل الأسلاك، وتعمل ضمن دائرة نصف قطرها 10 أمتار ، ويشيع استخدامها في الهواتف المحمولة.

1- بلوتوث هو بروتوكول شبكي، لذلك يمكنك العثور عليه تحت قائمة networking الرئيسية:

Networking
[*] Networking support
[*]   Bluetooth subsystem support

2- يوجد نوعان من بروتوكول البلوتوث للاختيار، كلاهما يجب عليك تفعيله كي تعمل مع كل أنوع أجهزة البلوتوث :

Networking
[*] Networking support

```
--- Bluetooth subsystem support
[*]   L2CAP protocol support
[*]   SCO links support
```

3- وهناك عدد قليل نسبيا من مشغلات أجهزة البلوتوث المتاحة، لأن أغلب هذه الأجهزة تتبع مواصفات بروتوكول بلوتوث بالتفصيل في كيفية عملها. هذه المشغلات عليها علامة في القائمة التالية يجب عليك اختيارها لجعل البلوتوث يعمل مع الجهاز :

```
Networking
   [*] Networking support
      --- Bluetooth subsystem support
                Bluetooth device drivers
               [M] HCI USB driver
               [*]   SCO (voice) support
               [ ] HCI UART driver
               [M] HCI BCM203x USB driver
               [M] HCI BPA10x USB driver
               [ ] HCI BlueFRITZ! USB driver
               [ ] HCI DTL1 (PC Card) driver
               [ ] HCI BT3C (PC Card) driver
               [ ] HCI BlueCard (PC Card) driver
               [ ] HCI UART (PC Card) device driver
               [ ] HCI VHCI (Virtual HCI device) driver
```

Wireless

شبكة Wireles تعتبر الأكثر شعبية مع أغلب أجهزة الحاسب المحمول والتي تحتوي على جهاز شبكة من نوع Wireles. وتدعم نواة لينكس قطاعا عريضا من مشغلات Wireles. مع المزيد منها يضاف كل أسبوع. وللتحقق من وجود جهاز PCI Wireles على نظامك وما هو نوعه اكتب الأمر التالي :

```
$ /usr/sbin/lspci | grep -i wireless
06:05.0 Network controller: Intel Corporation PRO/Wireless
2915ABG MiniPCI
Adapter (rev 05)
```

لاحظ أن الناتج عندك ربما يكون غير مطابق لهذا، ولكن المهم هو معرفة الأمر الذي يعرض لك بعضا من أجهزة PCI Wireles .

1- لتفعيل wireless support في لينكس ،يجب تفعيل الخيار IEEE 802.11

network configuration ، (والرقم 802.11 هو رقم مخصص للويرلس الذي تتبعه كل هذه الأجهزة):

Networking
  [*] Networking support
  [*]   Generic IEEE 802.11 Networking Stack

2- يجب أيضا تفعيل مختلف أنواع بروتوكول 802.11 و خيار Software MAC كي تحصل عل دعم كامل لجميع الأنواع المختلفة من أجهزة الوايرلس في لينكس :

Networking
  [*] Networking support
  [*]   Generic IEEE 802.11 Networking Stack
  [*]   IEEE 802.11 WEP encryption (802.1x)
  [M]   IEEE 802.11i CCMP support
  [M]   IEEE 802.11i TKIP encryption
  [M]   Software MAC add-on to the IEEE 802.11 networking stack

3-ا لدعم الخاص بمختلف أنواع أجهزة PCI wireless يوجد تحت قسم إعداد مشغل Network :

Device Drivers
  Network device support
    Wireless LAN (non-hamradio)
    [*]Wireless LAN drivers (non-hamradio) & Wireles Extensions
[*] Wireless Extension API over RtNetlink

هناك قطاع عريض من مشغلات أجهزة PCI اختر المناسب منها اعتماد على الجهاز الموجود لديك. مشغلات جهاز الشبكة من نوع USB wireless يوجد في قسم مختلف من الإعداد :

Device Drivers
  USB Support
    USB Network Adapters

Filesystems

تدعم نواة لينكس قطاعا عريضا من أنواع نظم الملفات التقليدية، وعددا من مختلف أنواع نظم الملفات (حجم مديري الملفات، ونظم الملفات العنقودية ، الخ).أنواع نظم

الملفات التقليدية (العادية أو journaled) يمكن اختيارها من القائمة الرئيسية لنظم الملفات داخل قائمة الإعداد :

File systems
    [*] Second extended fs support
    [*] Ext3 journalling file system support
    [ ] Reiserfs support
    [ ] JFS filesystem support
    [ ] XFS filesystem support

هذا القسم سوف يعرض لك بعض أنواع أنظمة الملفات التقليدية التي يدعمها لينكس وكيفية تفعيلها .

RAID

يقدم RAID خيارا للجمع بين العديد من الأقراص معا، ولذلك تبدو وكأنها قرص منطقي واحد . هذا يمكنه المساعدة في توفير طرق زائدة للإمداد أو السرعة عن طريق نشر البيانات عبر مختلف طبقات القرص الصلب. وتدعم نواة لينكس كلا من عتاد RAID وبرمجياته. عتاد RAID يتم التعامل معه عن طريق متحكم القرص الصلب disk controller، دون الحاجة إلى أي مساعدة من النواة.

1-برنامج RAID يتم التحكم به من خلال النواة ويمكن تحديده كخيار مدمج.

Device Drivers
    Multi-device support (RAID and LVM)
        [*] Multiple devices driver support (RAID and LVM)
        [*]   RAID support

2- هناك العديد من أنواع إعدادات RAID المختلفة، ويجب اختيار نوع واحد على الأقل كي يعمل RAID بشكل سليم.

Device Drivers
    Multi-device support (RAID and LVM)
        [*] Multiple devices driver support (RAID and LVM)
        [*]   RAID support
        [*]   Linear (append) mode
        [*]   RAID-0 (striping) mode
        [*]   RAID-1 (mirroring) mode
        [*]   RAID-10 (mirrored striping) mode (EXPERIMENTAL)
        [*]   RAID-4/RAID-5 mode
        [*]   RAID-6 mode

Logical Volume Manager and Device Mapper

Logical Volume Manager (LVM) يشبه RAID إلى حد كبير ، في أنه يسمح للمستخدم بالجمع بين أقراص مختلفة لتبدو وكأنها قرص منطقي واحد. على أية حال هو لا يعمل مع الأجهزة على نفس مستوى RAID، ولكن له آلية عمل من خلال مخطط الكتل والقطاعات التي تسمح لعدد مختلف من الأقراص الصلبة للعمل معا كوحدة واحدة لتبدو للمستخدم كأنها قرص كبير.

وللقيام بذلك تستخدم النواة شيئا ما يسمى Device Mapper (DM) :

1- قم بتفعيل DM support في النواة

```
Device Drivers
  Multi-device support (RAID and LVM)
    [*] Multiple devices driver support (RAID and LVM)
    [*]    Device mapper support
```

2- هناك عدد من الوحدات البرمجية modules المساعدة تعمل مع DM لتزوده بوظائف إضافية.يجب عليك تفعيلها إذا كنت ترغب في تشفير encrypt أقراصك أو إتاحة وظيفة snapshot :

```
Device Drivers
  Multi-device support (RAID and LVM)
    [*] Multiple devices driver support (RAID and LVM)
    [*]    Device mapper support
    [*]      Crypt target support
    [*]      Snapshot target (EXPERIMENTAL)
    [*]      Mirror target (EXPERIMENTAL)
    [*]      Zero target (EXPERIMENTAL)
    [*]      Multipath target (EXPERIMENTAL)
```

مشاركة الملفات مع ويندوز:

Samba برنامج يتيح لمستخدمي لينكس الوصول لأجهزة ويندوز محليا عبر الشبكة، ويقدم لنا طريقة لمشاركة الملفات والأجهزة بطريقة واضحة. وكذلك يتيح ل لينكس العمل كخادم ويندوز، متيحا لعملاء ويندوز الاتصال به معتقدين أنه جهاز ويندوز حقيقي .

هناك نوعان مختلفان من أنظمة الملفات تتيح لجهاز لينكس الاتصال مع جهاز ويندوز:

- نظام ملفات SMB
- نظام ملفات CIFS

ولقابلية الاتصال بأنظمة ويندوز القديمة ل Workgroups أو أجهزة ويندوز 95 أو ويندوز 98، قم بتحديد خيار SMB filesystem:

File systems
    Network File Systems
        [*] SMB file system support (to mount Windows shares etc.)

ومن أجل إمكانية الاتصال بأجهزة ويندوز الحديثة، فمن الموصى به تحديد خيار CIFS filesystem بدلا من SMB :

File systems
    Network File Systems
        [*] CIFS support

ولمزيد من التفاصيل حول الاختلافات ما بين هذين النوعين من أنظمة الملفات ، ومتى يجب عليك استخدام أحدهما بدلا من الآخر، يرجى مطالعة الخانة الخاصة بـ SMB_FS و CIFS في الفصل 11

OCFS2

OCFS2 هو نظام ملفات عنقودي من شركة أوراكل يعمل على تثبيت أنظمة شبكات محلية كبيرة و صغيرة في آن واحد .هذا النوع من نظم الملفات يوصى به عند استخدام قواعد بيانات كبيرة،مثل Oracle و DB2، لأنه يمكن نقلها مع مرور الوقت لمختلف الأقراص عبر الشبكة بسهولة كبيرة كلما كانت هناك حاجة للمزيد من مساحة التخزين .

لتفعيل نظام الملفات OCFS2 :

File systems
    [*] OCFS2 file system support

Security

تدعم نواة لينكس نماذج لمختلف نظم الأمن عن طريق تقديم الاستحكامات، وتتيح لك تبني النموذج الخاص بك حسب اختيارك.وحاليا يوجد القليل من النماذج الافتراضية التي تأتي مع ملف مصدر نواة لينكس . ولكن مطوري النماذج الحديثة يعملون على أن تنال الكثير من الرضا.

Default Linux Capabilities

النوع القياسي لنموذج الأمن في لينكس هو "capability".يجب عليك دائما أن تحدد هذا الخيار إلا إذا كنت بالفعل تريد تشغيل نواة غير آمنة لسبب ما .ولتفعيل ذلك:

Security options
    [*] Enable different security models
    [*]   Default Linux Capabilities

SELinux:

أكثر النظم الأمنية شعبية يسمى SELinux . هذا النموذج مدعوم من العديد من توزيعات لينكس.

SELinux يتطلب تفعيل خيار networking.انظر القسم السابق "networking" لتفعيله.

SELinux يتطلب أيضا تفعيل Auditing support أثناء إعداد النواة. ولفعل ذلك:

General setup
    [*] Auditing support

كذلك يجب تفعيل خيار networking security :

security options
   [*] Enable different security models
   [*]   Socket and Networking Security Hooks

والآن يمكنك تحديد خيار SELinux :

Security options
    [*] Enable different security models
    [*] NSA SELinux Support

وهناك أيضا عدد من الخيارات الفردية لـ SELinux التي قد ترغب في تفعيلها. يرجى الاطلاع على ملفات المساعدة ، لمعرفة المزيد من التفاصيل عما تقوم به هذه العناصر المختلفة :

Security options
    [*] Enable different security models
    [*] NSA SELinux Support
    [ ]   NSA SELinux boot parameter
    [ ]   NSA SELinux runtime disable

[*] NSA SELinux Development Support
[*] NSA SELinux AVC Statistics
NSA SELinux checkreqprot default value

Kernel Debugging:

هناك مجموعة واسعة من الخيارات المختلفة للنواة يمكن أن تساعد في تصحيح أخطاء ما يجري داخل النواة. وفيما يلي قائمة لبعض المشاكل الأكثر شيوعا التي يمكن أن تفيد في اكتشاف أشياء جديدة حول كيفية عمل النواة، أو المساعدة على إيجاد المشاكل المحتملة داخل شفرة الملف المصدري للنواة الحالية .

Kernel Log Timestamps

تنتج النواة طائفة واسعة من الرسائل إلى سجل الدخول logfile الخاص بها ،هذه الرسائل يمكن رؤيتها عن طريق الاطلاع على logfile الخاص بالنظام (عادة يوجد في المسار */var/log/messages/*) أو من خلال تشغيل الأمر *dmesg* .
في بعض الأحيان يكون من المفيد أن نرى بالضبط متى صدرت هذه الرسائل. على كل حال *dmesg* لا يضع أي بصمة زمنية-timestamps على الأحداث التي تظهر ، ودقة الوقت في الملف */var/log/messages* فقط لأقرب ثانية. يمكنك تهيئة النواة الخصيص كل رسالة بطابع زمني لأقصى دقة تصل إليها النواة في قياس قيمة الوقت (عادة المدى الزمني هو بالميكرو ثانية).
ولتفعيل خيار البصمة الزمنية timestamp على رسائل النواة :
Kernel hacking
[*] Show timing information on printks

Magic SysRq Keys

مفتاح SysRq على لوحة المفاتيح يمكن استخدامه للتحكم في النواة بعدة طرق مختلفة أثناء عمل النواة، أو بعد تحطمها.
لتفعيل هذا الخيار :
kernel hacking
[*] Magic SysRq
لوصف كامل عن الأحداث المختلفة التي يمكن إحداثها عن طريق هذا الخيار يرجى مطالعة ملف الوثائق
Documentation/sysrq.txt في شجرة الملف المصدري للنواة.

:Debug Filesystem

نظام الملفات الأساسي في الذاكرة العشوائية يمكن استخدامه في الخرج الخاص بالعديد من معلومات تصحيح الأخطاء debugging.نظام الملفات هذا يسمى debugfs ويمكن تفعيله كما يلي:

Kernel hacking
  [*] Debug filesystem

بعد تفعيلك لهذا الخيار والإقلاع بالنواة التي أعيد بناؤها، يقوم بإنشاء هذا المسار */sys/kernel/debug* كموضع يستخدمه المستخدم لربطه مع نظام ملفات debugfs ، ولعمل ذلك يدويا اكتب:

```
$ mount -t debugfs none /sys/kernel/debug
```

ويمكن الحصول على هذا النوع من نظام الملفات ليتم ربطه آلياً وقت الإقلاع عن طريق إضافة هذا السطر إلى الملف */etc/fstab* :

```
debugfs /sys/kernel/debug debugfs 0 0
```

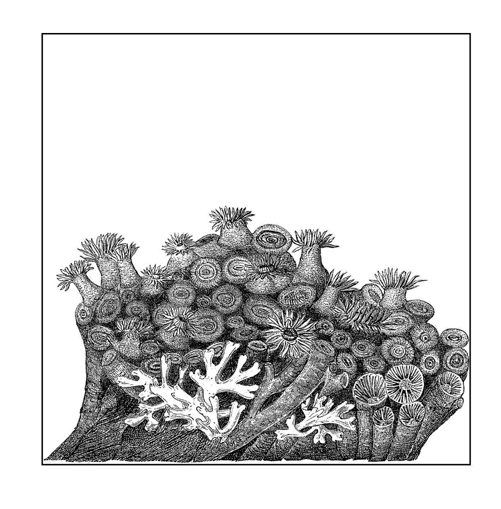
بعد توصيل *debugfs*، سوف يتحول عدد كبير من مختلف الملفات والأدلة إلى الدليل */sys/kernel/debug/* وكل ذلك يتم تشغيله بشكل افتراضي وديناميكي عن طريق النواة ، مثل الملفات الموجودة في *procfs* أو *sysfs*.،هذه الملفات يمكن استخدامها لمساعدة تصحيح الأخطاء لمختلف النظم الفرعية في النواة ، أو فقط لمراقبة ماذا يحدث للنظام .

General Kernel Debugging

وهنا مجموعة أخرى من الخيارات الجيدة لإعداد النواة التي قد ترغب في تمكينها إذا كنت تريد مساعدة مطوري النواة في تصحيح مشاكل مختلفة ، أو فقط معرفة المزيد عن كيفية عمل النواة من خلال النظر إلى هذه الرسائل التي تنتجها لنا هذه الخيارات . علما أنه اذا قمت بتفعيل أغلب هذه الخيارات ، فإن النواة سوف تكون أبطأ بمقدار قليل. لذلك إذا لاحظت أي نقص في الأداء، فقد ترغب في تعطيل هذه الخيارات :

Kernel hacking
  [*] Kernel debugging
  [*]   Detect Soft Lockups
  [ ]   Collect scheduler statistics
  [*]   Debug slab memory allocations
  [*]   Memory leak debugging
  [*]   Mutex debugging, deadlock detection
  [*]   Spinlock debugging

[*]   Sleep-inside-spinlock checking
[ ]   kobject debugging
[ ]   Highmem debugging
[ ]   Compile the kernel with debug info



9

مرجع لمعاملات أوامر إقلاع النواة

أغلب هذا الفصل يقوم على أساس الوثائق الداخلية
النواة لشرح دلالة مختلف أنواع خيارات سطر الأوامر الخاص بإقلاع النواة، والتي كتبها
مطورو النواة والصادرة بموجب رخصة GPL .
هناك ثلاثة طرق لتمرير الخيارات إلى النواة، وبالتالي التحكم في سلوكها :

- عند بناء النواة، وأغلب مادة هذا الكتاب تناقش هذه الخيارات.
- عند إقلاع النواة. وعادة يتم تمرير معاملات إلى النواة ، ويتم استحضار هذه المعاملات من ملف إقلاع مثل ملف التهيئة الخاص بـ GRUB أو LILO .
- وقت التشغيل ،عن طريق الكتابة إلى الملفات الموجودة في المسارات */proc* و /sys .

هذا الفصل يشرح الطريقة الثانية الخاصة بتمرير الخيارات.
يزيل هذا الفصل الغموض المحيط بخيارات وقت الإقلاع من خلال عدة أجزاء منطقية. رقم تحديد معمارية المعالج وخيارات مشغل محدد لا تندرج في القائمة هنا.
وللحصول على قائمة كاملة لكل الخيارات المعروفة، يرجى الاطلاع على ملف *Documentation/kernel-parameters.txt* في شجرة الملف المصدري للنواة ، والملفات المفردة لوثائق كل معمارية مخصصة على حدة.
ليست كل الخيارات المدرجة في القائمة متاحة دائما. وأغلبها مشترك مع النظم الفرعية، ويعمل فقط إذا تم تهيئة النواة مع هذه النظم الفرعية المدمجة بها.
وهي تعتمد أيضا على وجود العتاد الذي ينضوي تحتها.
كل هذه المعاملات حساسة لحالة الأحرف case-sensitive.

خيارات تخصيص الموديل Module-Specific Options

بالإضافة إلى الخيارات المدرجة بالقائمة في هذا الفصل، يمكن أيضا تمرير معاملات ال modules المبنية داخل النواة على سطر الأوامر.

(وبالطبع فإن الموديلز المحملة لا تتواجد في الذاكرة في أوقات إقلاع النواة، ولذلك لا يمكن تمريرها كمعاملات عند الإقلاع.)

تتألف صيغة المعاملات من اسم الموديل متبوعة بنقطة (.) ثم المعامل.

على سبيل المثال ؛ الموديل *usbcore* قبل المعامل blinkenlights لتشغيل ضوء الوميض على كل أجهزة USB 2.0 hubs المدعومة (لا تقل أبدا إن مطوري النواة يفتقدون لروح الدعابة). لوضع هذا المعامل عند تحميل هذا الموديل ديناميكيا ، عليك أن تكتب :

```
$ modprobe usbcore blinkenlights=1
```

ولكن إذا كان الموديل *usbcore* مبنيا بداخل النواة ، فسوف تنجز نفس العمل عن طريق تنفيذ النواة للخيار التالي :

```
usbcore.blinkenlights=1
```

معظم خيارات ال module الخاصة بال modules المبنية داخل النواة يمكن أيضا تغييرها وقت التشغيل عن طريق الكتابة إلى المجلدات الفرعية المذكورة بعد اسم الموديل تحت المسار */sys/module*. وبالتالي يتم استحضار الخيار *blinkenlights* عن طريق الملف *sys/module/usbcore/blinkenlights*.

Console Options

هذه الخيارات التي تتعامل مع الكونسول أو kernel log ، وهو المكان الذي تعرض من خلاله ال debugging والمعلومات المتعلقة بالأخطاء.

console	Output console device and options. console=Options ttyn استخدم جهاز الكونسول التخيلي n. ttySn[,*options*], ttyUSB0[,*options*] استخدم منفذ سيريال معين. الخيارات التي لها الصيغة *bbbbpnf* ، حيث *bbbb* هي baud rate ، و *p* تعادل(n,o,أو e) ، *n* هي عدد البايتات و *f* هي flow control- مراقبة الدفق r) ل RTS أو يتم إهماله) القيمة الافتراضية هي 9600n8. لمزيد من التفاصيل عن كيفية استخدام serial console انظر *Documentation/serial-console.txt* ..

	إذا كنت ترغب في الوصول إلى معلومات ال serial console وليس لديك منفذ سيريال ، اقرأ خيار سطر الأوامر *netconsole* . uart,io,*addr*[,*options*], uart,mmio,*addr*[,*options*] شغل مسبقا كونسول من نوع *polled-mode* على منفذ UART 8250/16550 لمنفذ I/O مخصص أو عنوان MMIO[1] والتبديل إلى جهاز ttyS لاحقا . هذه الخيارات هي نفسهاالخاصة ب ttyS والمعروضة آنفا .
netconsole	Output console data across the network. netconsole=[*src-port*]@[*src-ip*]/[*dev*],[*target-port*]@*target-ip*/[*target- mac-address*] أرسل بيانات كونسول النواة عبر الشبكة باستخدام حزم UDP إلى جهاز آخر. الخيارات هي : *src-port* منفذ مصدري لحزم UDP. القيمة الافتراضية هي 6665. *src-ip* عنوان Ip المصدر الذي تستخدمه واجهة الشبكة. *dev* واجهة(بطاقة) الشبكة المستخدمة.eth0 على سبيل المثال. واجهة الشبكة يمكنها أيضا تشغيل حركة سير طبيعي للشبكة. لأن بيانات ال netconsole ليست متداخلة وينبغي ألا تسبب بطئا في العمليات الأخرى للشبكة. *Target-port* المنفذ الذي سيستخدمه عميل التسجيل logging agent . *Target-mac-address* العنوان الفيزيائي لبطاقة شبكة logging agent. للإنصات لهذه البيانات يمكن للجهاز البعيد استخدام برنامج *syslogd* ، أو تشغيل برنامج *netcat* كما يلي: netcat -u -l -p port ولمزيد من الخلفية عن كيفية استخدام هذا الخيار، انظر الملف *Documentation/networking/netconsole.txt*.
debug	Enable kernel debugging. يجعل مستوى السجل الخاص بالنواة مضبوطا على debug level،

(1) **Memory-mapped I/O : MMIO**

	حيث ستظهر كل رسائل ال debug على الكونسول أثناء الإقلاع.
quiet	Disable all log messages. ضبط مستوى السجل للنواة على KERN_WARNING (4)، والذي يمنع كل الرسائل أثناء الإقلاع، فيما عدا الخطير للغاية منها.(Log levels سيتم شرحها تحت المعامل *loglevel*)
earlyprintk	Show early boot messages. `earlyprintk=[vga|serial][,ttySn[,baudrate]][,keep]` يعرض رسائل سجل النواة التي تسبق بداية الكونسول التقليدي. هذه الرسائل لا تظهر تماما على الكونسول إلا إذا استخدمت هذا الخيار. وتفعيله يمكن أن يكون مفيدا جدا فيما يتعلق بتخفيض تعقب العتاد. وحاليا هذا الخيار يحدد إما بطاقة VGA وإما منفذ السيريال، ولكن ليس كلاهما معا.كذلك ، جهاز سيريال ttyS0 أو ttyS1 فقط سوف يعمل. التفاعل مع مشغل السيريال الافتراضي ليس بالأمر الجيد، وخرج ال VGA سيتم الكتابة فوقه بشكل نهائي من خلال الكونسول الفعلي.إضافة إلى استمرار منع الرسائل المعروضة بهذا الخيار حتى يبدأ الكونسول الفعلي للنواة في تولي أمر النظام.
loglevel	Set the default console log level. `loglevel=level` يحدد مستوى السجل log level الخاص بالكونسول الأساسي . أي رسائل للسجل مع مستويات أقل من هذا (الأولوية الأعلى) سوف يتم عرضها على الكونسول، حيث إن أي رسائل مع مستويات تساوي أو تزيد عن هذا فلن يتم عرضها . ويمكن أيضا تغيير console log level عن طريق برنامج *klogd*، أو عن طريق كتابة المستوى المعين إلى الملف */proc/sys/kernel/printk* . ومستويات التسجيل الخاصة بالنواة هي: **(KERN_EMERG) 0** النظام غير مستخدم. **(KERN_ALERT) 1** الأحداث التي يجب أخذها بعين الاعتبار في الحال. **(KERN_CRIT) 2** حالات حرجة - Critical conditions.

	3 (KERN_ERR) حالات خطأ غير حرجة - Noncritical error conditions. 4 (KERN_WARNING) حالات تحذير يجب أن تؤخذ بعناية. 5 (KERN_NOTICE) أحداث عادية ولكن مهمة. 6 (KERN_INFO) رسائل معلومات لا تتطلب عملا ما . 7 (KERN_DEBUG) رسائل debugging للنواة، الصادرة من النواة إذا قام المطور بتفعيل ال debugging وقت الكومبايل.
log_buf_len	Set the size of the kernel log buffer . `log_buf_len=n[KMG]` يضبط حجم البفر الأساسي الخاص بسجل النواة. *n* يجب أن تكون رقم 2 مرفوعا بأي أس - power of 2، فإذا لم يكن كذلك ستكون قريبا من power of 2. هذه القيمة يمكن أيضا تغييرها عن طريق قيمة تهيئة للنواة : CONFIG_LOG_BUF_SHIFT
initcall_debug	Debug the initcall functions in the kernel. يجعل النواة تتبع كل الدوال المستدعاة من النواة أثناء ابتداء النظام كما يحدث في إقلاعات النواة. هذا الخيار تحديده مفيد عندما يقع موت للنواة أثناء الإقلاع.
kstack	How many words of the stack to print in kernel oopses. `kstack=n` يحدد عدد الكلمات من مكدس النواة kernel stack التي ينبغي طباعته في kernel oops dumps ، و *n* قيمة ذات عدد صحيح.
time	Show timing data on every kernel log message تجعل النواة تصدر كل رسالة لسجل النواة مع الطابع الزمني timestamp. الخاص بها.

خيارات المقاطعة- Interrupt Options

المقاطعات هي جانب معقدة من جوانب سلوك النواة. خيارات وقت الإقلاع تعامل في الغالب مع الواجهة بين النواة وبين العتاد الذي يتعامل بهذه المقاطعات.مثل متحكم المقاطعة المتقدم القابل للبرمجة Advanced Programmable Interrupt

Controller (APIC . الخاص برقاقة إنتل.

apic	Change the verbosity of the APIC subsystem when booting. `apic=[quiet|verbose|debug]` يتحكم في كم المعلومات التي يولدها النظام الفرعي APIC عند إقلاع النواة. الوضع الافتراضي هو quiet.
noapic	Do not use any IOAPICs. يمنع النواة من استخدام أي نوع من دخل/خرج المتحكمات القابلة للبرمجة IOAPICs [(1)] التي ربما تكون موجودة في النظام
lapic	Enable the local APIC. يجعل النواة تفعل APIC المحلي حتى في حالة تعطيل البيوس لها.
nolapic	Do not use the local APIC. أبلاغ النواة بألا تستخدم local APIC.
noirqbalance	Disable kernel IRQ balancing. تعطيل منطق التوازن لكل طلبات المقاطعة المدمجة بالنواة .
irqfixup	Basic fix to interrupt problems. عندما لا يتم معالجة أحد طلبات المقاطعة ، ابحث عن جميع معالجات المقاطعة الخاص به. وذلك بقصد الحصول على أنظمة تعمل مع برامج ثابتة للعتاد - firmware - رديئة.
irqpoll	Extended fix to interrupt problems. عندما لا يتم معالجة أحد طلبات المقاطعة ، ابحث عن جميع معالجات المقاطعة الخاص به، وكذلك قم بفحص كل معالجات المقاطعة لكل مؤقت المقاطعة timer interrupt. وذلك بقصد الحصول على نظم تعمل على برامج firmware رديئة.

(1)انظر لمزيد من المعلومات
http://www.intel.com/design/chipsets/datashts/290566.htm
و http://en.wikipedia.org/wiki/Intel_APIC_Architecture .

noirqdebug	Disable unhandled interrupt detection. افتراضيا ، تحاول النواة الكشف عن مصدر المقاطعة غير المعالجة وتعطيله، لأنه يمكن أن يسبب مشاكل بخصوص استجابة بقية النواة إذا لم يتم إيقافه. هذا الخيار لتعطيل هذا المنطق.

Memory Options - خيارات الذاكرة

تتعامل النواة مع الذاكرة في العديد من الأجزاء والفئات لأغراض مختلفة . هذه الخيارات تتيح لك تعديل الأحجام والإعدادات .

highmem	Specify the size of the highmem memory zone. `highmem=n` فرض نطاق ذاكرة highmem للحصول على الحجم الفعلي ل *n* من البايتات.وذلك سوف يعمل حتى على الأنظمة التي ليس لها highmem zone افتراضيا . ويمكن كذلك تقليل حجم نطاق ذاكرة highmem zone للأجهزة التي عليها الكثير من الذاكرة.
hugepages	Set the number of hugetlb pages. `hugepages=n` hugetlb ميزة تتيح لك تهيئة لينكس لاستخدام MB 4 صفحة، ألف مرة من مقدار الحجم الافتراضي .إذا تم تهيئة النواة بهذا الشكل، فهذه الخيارات تضع الحد الأقصى من عدد صفحات hugetlb لتكون *n* .
ihash_entries	Set the number of inode hash buckets `ihash_entries=n` تجاوز العدد الافتراضي من حاويات التلبيد hash buckets لعقد تخزين النواة kernel's inode cache ، ينصح به لخبراء النواة فحسب .
max_addr	Ignore memory. `max_addr=n` يجعل النواة تتجاهل جميع الذاكرة الفيزيائية التي أكبر من أو تساوي العنوان الفيزيائي *n* .

mem	Force memory usage. `mem=n[KMG]` وضع مقدار معين من الذاكرة المستخدمة من النواة.عندما تستخدم مع memmap= option، يمكن تجنب التعارضات بين أماكن العناوين الفيزيائية. بدون memmap= option، هذا الخيار يمكن أن يؤدي إلى وضع أجهزة PCI على عناوين تنتمي إلى عناوين غير مستخدمة في الذاكرة. *n* تحدد مقدار الذاكرة التي تفرض، وتقاس بوحدات الكيلوبايت(K)، والميجابايت (M)، أو بالجيجابايت (G)
mem	Disable the use of 4 MB pages for kernel memory. `mem=nopentium` يعطل استخدام الصفحات الكبيرة (4 MB) لذاكرة النواة.
memmap	Enable setting of an exact E820 memory map. `memmap=exactmap` *استخدام مخطط مخصوص للذاكرة memory map*.خطوط *exactmap* يمكن إنشاؤها على أساس خرج البيوس أو متطلبات أخرى
memmap	Force specific memory to be used. `memmap=n[KMG]@start[KMG]` إجبار النواة على استعمال نطاق محدد من الذاكرة. *n* هو حجم الذاكرة، و *start* معناها مكان البداية الخاص بنطاق الذاكرة، ويمكن قياسها بوحدات الكيلوبايت(K)، والميجابايت (M)، أو بالجيجابايت (G)
noexec	Enable or disable nonexecutable mappings. `noexec=[on|off]` فعل أو عطل قدرة النواة لرسم مخطط لأقسام النواة ك nonexecutable . وافتراضيا يكون خاصية mapping مفعلة .(on)
reserve	Reserve some I/O memory. `reserve=n[KMG]` أجبر النواة على تجاهل بعضا من دخل/خرج I/O أجزاء النواة.
vmalloc	Force the vmalloc area to have a specific size. `vmalloc=n[KMG]`

	أجبر *vmalloc* للحصول على الحجم الفعلي المحدد ب *n* .وذلك يمكن استخدامه لزيادة الحد الأدنى لحجم مساحة *vmalloc* (التي تكون MB 128 على معالجات x86). ويمكن أيضا استخدامها لتقليل الحجم وإخلاء المزيد من الغرف للذاكرة المخططة على النواة mapped kernel RAM .
norandmaps	Do not use address space randomization. افتراضيا تقوم النواة عشوائيا بترتيب أماكن عناوين جميع البرامج عند بداية عملها. هذا الخيار يعطل هذه السمة . وهذا يساوي كتابة 0 للملف */proc/sys/kernel/randomize_va_space* .
vdso	Enable or disable the VDSO mapping. vdso=[0|1] عطل (0) او فعل (1) خيار التخطيط VDSO (Virtual Dynamic Object Shared). هذا الخيار مفعل افتراضيا.

خيارات النوم Suspend Options

هذه الخيارات تغير الطريقة التي تتعامل بها النواة مع وضع النوم بقصد توفير الطاقة.

---

resume

Specify the partition device for the suspend image.

resume=suspend_device

أخبر النواة بالقرص الذي يحتوي على صورة عملية النوام الخاصة بالنواة suspended kernel image

إذا كانت البيانات على الصورة بيانات صحيحة أنشئت من قبل النظام الفرعي للبرنامج الخاص بحالة النوم ، فسيتم تحميله إلى الذاكرة وسوف تقوم النواة بتشغيله بدلا من مواصلة عملية إقلاع عادية.

suspend_device هو اسم القرص على النواة الذي ربما يكون مختلفا عن الاسم الذي يستخدمه حيز المستخدم userspace ، لذلك كن حذرا مع هذا الخيار.

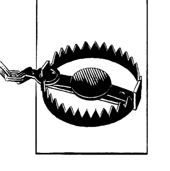
noresume

Disable resume.

عطلة وظيفة الاستعادة resume الخاصة بالنواة. أي أقسام سواب كانت مستخدمة لحمل صور النظام التي تستخدمها النواة لعملية الاسترجاع سوف تعود وتضاف إلى المساحة المتاحة من السواب.

خيارات المعالج CPU Options

هذه الخيارات تتحكم في قطاع عريض من السلوكيات، بشأن التوقيت، واستخدام المعالج في النظم متعددة المعالج ، وغيرها من الأمور.

cachesize	Override level 2 CPU cache size detection. `cachesize=n` أحيانا تقوم أخطاء bugs أجهزة المعالج بعمل تقرير عن حجم المخبأ cache بشكل خاطئ. ستحاول النواة العمل على إصلاح الأخطاء المعروفة مع أغلب المعالجات، ولكن لا يمكن مع أغلب المعالجات تحديد الحجم الصحيح .هذا الخيار يساعد على تجاوز هذه المواقف ويقاس *n* بالبايت .
lpj	.Set the loops per jiffy `lpg=n` حدد الحلقات بال *jiffy* (1) التي يجب على النواة استخدامها. وبذلك تتجنب النواة التحديد التلقائي للوقت المستهلك في الإقلاع بهذه القيمة. إذا كانت *n* تساوي 0 ، سيتم التعرف التلقائي لهذه القيمة كالمعتاد.  بالنسبة للأنظمة متعددة المعالج SMP سيتم ضبط هذه القيمة على جميع المعالجات، حيث يمكن أن تسبب المشاكل إذا كانت المعالجات المختلفة تحتاج إلى إعدادات مختلفة. أي قيمة خاطئة سوف تؤدي إلى نسب تأخير خاطئة في النواة ، مما يؤدي بدوره إلى أخطاء I/O لا يمكن التنبؤ بها وغيرها من الخسائر.على الرغم من الاحتمال البعيد في الحالات القصوى أن ذلك قد يؤدي إلى تلف العتاد الخاص بك.
nmi_watchdog	Set the NMI watchdog value. `nmi_watchdog=[0|1|2|3]` هذه ميزة تصحيح الأخطاء debugging التي تتيح لك تجاوز قيمة المقاطعة غير المقنعة الافتراضية (nonmaskable interrupt watchdog (NMI (1)

(1) jiffie مصطلح معناه الحرفي "كلمح البصر" وهي دلالة على مقدار زمني غاية في القصر وتستخدم في تطبيقات مختلفة كفترات قصيرة من الوقت ؛ وفي الحوسبة : jiffie هي الفترة الزمنية التي تستغرقها دورة مقاطعة النظام وهي ليست وحدة زمنية مطلقة حيث إنها تعتمد على دقة تردد ساعة المقاطعة الخاص بالعتاد . عادة ، هذا الوقت هو 0.01 ثانية.

	القيمة 0 تعني لا تستخدم NMI watchdog.و 1 تعني استخدام APIC في حالة وجوده. و 2 تعني أنه يجب استخدام local APIC إذا كان موجودا . و 3 تعني أن NMI watchdog صالحة، لذا لا تستخدمها.
no387	Always use the 387 emulation library استخدم دائما مكتبة المحاكاة 387 بعيدة الاحتمال، حتى ولو كانت 387 math coprocessor موجودة على النظام.
nofxsr	Disable x86 floating-point save and restore. عطل استعادة وحفظ سجل النقطة العائمة الممتد لمعالجات x86 . ستقوم النواة فقط بحفظ سجلات النقاط العائمة القديمة على task switch .
no-hlt	Do not use the HLT instruction. هذا الخيار متاح لأن التعليمة HLT لا تعمل بشكل صحيح على معالجات x86. هذا الخيار يبلغ النواة بعدم استخدام هذه التعليمة.
mce	Enable the machine check exception feature. بعض المعالجات تستطيع فحص أخطاء الأجهزة (الأخطاء المعتادة في العتاد). هذا الخيار يشغل هذا النظام الفرعي. إذا تم بناؤه داخل تهيئة النواة.
nomce	Disable the machine check exception feature. هذا الخيار يوقف النظام الفرعي ل mce.
nosep	Disable x86 SYSENTER/SYSEXIT support. أبطل دعم x86 SYSENTER/SYSEXIT في النواة.وذلك يمكنه جعل بعض نداءات النظام تأخذ وقتا أطول.
nosmp	Run as a single-processor machine. إبلاغ النواة متعددة المعالجات SMP لتعمل وكأنها نواة وحيدة المعالج ن حتى ولو على جهازمتعدد المعالجات.
notsc	Disable the time stamp counter. إبطال جهاز عداد البصمة الزمنية على النظام، إن كان موجودا.

Watchdog(1): هو جهاز أو متحكم إلكتروني يعمل على مراقبة التوقيت على الحاسب ويعمل على إعادة تشغيله إذا حدث تعليق لبرنامج رئيسي أو إهمال لإحدى الخدمات النظامية .. للمزيد من التفاصيل انظر:
http://en.wikipedia.org/wiki/Watchdog_timer

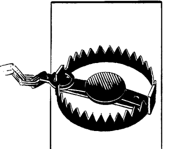
max_cpus	Maximum number of CPUs to use. maxcpus=*n* يحدد الحد الأقصى للمعالجات التي ينبغي للنواة متعددة المعالجات SMP استخدامها، حتى لو كان هناك المزيد من المعالجات موجودة على النظام.

خيارات الجدولة Scheduler Options

هذه الخيارات تعدل المعاملات المستخدمة في أحكام الجدولة scheduling decisions.
أغلبها يعتمد على فهم طريقة عمل الجدولة في لينكس .

---

isolcpus	Isolate CPUs from the kernel scheduler. isolcpus=cpu_number[,cpu_number,...] أزل المعالجات المعينة، التي تم تحديدها بقيم cpu_number، من الحسابات العامة للجدولة والاتزان الخاصة بالنواة متعددة المعالجات. الطريقة الوحيدة لإزالة أو تشغيل معالج معزول هو عن طريق نداءات النظام ذات الصلة. هذا الخيار وسيلة مميزة لعزل المعالجات.اختياريا الإعداد اليدوي لقناع المعالج في جميع المهام في النظام يمكن أن يسبب المشكلات وأداء غير مثالي في توازن توزيع عبء العمل .
migration_cost	Override the default scheduler migrations costs. migration_cost=level-1-useconds[level-2-useconds...] هذا خيار تصحيح الأخطاء الذي يتجاوزالجدول الافتراضي لمصفوفة تكلفة الإزاحة migration_cost. الأعداد المحددة في level-N-useconds مفهرسة من خلال "المجال المضيف للمعالجة"وتقاس بالميكروثانية. وكمثال على هذا الخيار migration_cost=1000,2000,3000 خاص بجهاز به خاصية SMT NUMA. وتلك تنشئ تكلفة أساسية للإزاحة ل 1 ميللي ثانية ms، وآخر تكلفة أساسية للإزاحة ل 2 ميللي ثانية ms ، وآخر تكلفة أساسية للإزاحة ل 3 ميللي ثانية ms . القيم غير الصحيحة يمكن أن يخفض أداء الجدولة بشكل شديد، حتى هذا الخيار ينبغي أن يستخدم فقط في

	تطوير الجدولة، وليس لبيئات الإنتاج أبدا .
migration_debug	Verbosity of migration cost autodetection. migration_debug=[0|1|2] ضبط مستوى تصحيح تكلفة الإزاحة. إذا تم وضع 0، لا يتم عرض المزيد من الرسائل في سجل النواة.هذه هي القيمة الافتراضية. 1 يطبع بعض المعلومات عن كيفية تحديد المصفوفة. 2 أكثر حوارية ومفيد فقط إذا كنت تستخدم serial console، مثل هذا القدر من المعلومات يسبب الحمل الزائد على البفر الخاص بولوج النواة.
migration_factor	Multiply or divide the migration costs. migration_factor=percent تعديل التكاليف الافتراضية للإزاحة بنسبة مئوية محددة. هذا هو خيار التصحيح الذي يمكن استخدامه تناسبيا في زيادة أو تخفيض التعرف التلقائي لتكاليف الإزاحة لجميع خانات مصفوفات الإزاحة التي تعرف تلقائيا .على سبيل المثال،migration_factor=150 تزيد تكاليف الإزاحة بنسبة 50%، حيث إن المجدول أقل تحمسا لمهام إزاحة cache-hot. المثال migration_factor=80 يقل تكاليف الإزاحة بمقدار 20%، وذلك يجعل المجدول أكثر حماسا لمهام الإزاحة.  القيم غير الصحيحة يمكن أن يخفض أداء الجدولة بشكل شديد، حتى هذا الخيار ينبغي أن يستخدم فقط في تطوير الجدولة، وليس لبيئات الإنتاج أبدا .

خيارات قرص الذاكرة Ramdisk Options

هذه الخيارات تتحكم في كيفية عمل مخزن المعلومات في الذاكرة المستخدمة شبيه بالأقراص (Ramdisk Options). إضافة إلى بدء عمل أقراص الذاكرة التي تحمل المعلومات الضرورية لبعض مراحل الإقلاع.

initrd	Location of initial ramdisk. initrd=filename حدد المكان الأولي لقرص الذاكرة اللازم لإقلاع النواة.

load_ramdisk	.oad a kernel ramdisk from a floppy load_ramdisk=n إذا كانت *n* تساوي 1، سيتم تحميل قرص الذاكرة من خلال النواة أثناء الإقلاع من محرك القرص المرن.
noinitrd	.Do not use any initrd لا تستخدم أي قرص ذاكرة أساسي ، حتى ولو تم تهيئته في خيارات أخرى ممررة إلى النواة.
prompt_ ramdisk	Prompt for the list of ramdisks. prompt_ramdisk=1 قم بإعلام المستخدم بقرص الذاكرة الأولي قبل الإعلام بقراءته من محرك القرص المرن.
ramdisk_ blocksize	Blocksize of the ramdisk. ramdisk_blocksize=n أبلغ مشغل قرص الذاكرة بمقدار البايتات التي يستخدمه لكل block. الحجم الافتراضي هو 1,024.
ramdisk_size	ize of the ramdisk. ramdisk_size=n حدد حجم قرص الذاكرة الأولي بالكيلوبايت. الحجم الافتراضي هو 4,096 (4 MB). هذا الخارينبغي استبداله بدلا من خيار سطر الأوامر الأقدم ل *ramdisk* .

Root Disk Options خيارات القرص الجذر

هذه الخيارات تتحكم في كيفية عثور وتعامل النواة مع نظم الملفات التي تحتوي على نظام ملفات الجذر.

ro	Mount the root device read-only on boot. الوضع الافتراضي للنواة هو عمل ماونت لقرص الجذر في نمط القراءة فقط read-only أثناء الإقلاع. هذا الخيار يتأكد من أن ذلك هو الذي تستخدمه النواة. وذلك يلغي خيار سطر الأوامر rw، إذا تم تعيينه مسبقا في سطر أوامر الإقلاع.

<table>
<tr><td>root</td><td>Specify the root filesystem to boot from.
root=device
أبلغ النواة ما هو القرص الذي يوجد عليه نظام ملفات القرص. Device يمكن تحديده بواحدة من الطرق التالية:
nnnn
هو رقم القرص بالنظام السداس عشري hexadecimal يمثل الرقم الرئيس major والرقم الثانوي minor للقرص في الصيغة الداخلية التي تتوقعها النواة .هذه الطريقة لا ينصح بها ما لم تكن وصلت إلى دواخل النواة .
/dev/nfs
استخدم قرص NFS محدد من خلال خيار الإقلاع nfsroot كقرص للجذر.
/dev/<diskname>
استخدم اسم قرص النواة المحدد ب <diskname> كقرص جذر.
/dev/<diskname><decimal>
استخدم اسم قرص النواة المحدد ب <diskname> والبارتيشن المحدد ب <decimal> كقرص جذر.
/dev/<diskname>p<decimal>
استخدم اسم قرص النواة المحدد ب <diskname> والبارتيشن المحدد ب <decimal> كقرص جذر. وهذا هو نفسه المذكور أعلاه، ولكنه لازم عندما يكون <diskname> ينتهي برقم.</td></tr>
<tr><td>rootdelay</td><td>Time to delay before attempting to mount the root filesystem.
rootdelay=n
انتظر n من الثواني قبل محاولة عمل ماونت لنظام ملفات الجذر. وذلك يمكن الاستفادة منه إذا كان نظام ملفات الجذر على قرص من نوع USB أو FireWire، حيث إن هذه الأقراص تأخذ وقتا أطول بقليل كي تكتشفها النواة.</td></tr>
<tr><td>rootflags</td><td>The root filesystem mount options.
rootflags=options
خيارات الوصل Mount التي ينبغي على النواة استخدامها في ربط نظام ملفات الجذر. القيمة options تتوقف على نوع نظام الملفات؛انظر الوثائق المتعلقة بالأنواع المخصصة لمعرفة التفاصيل عن الصالح منها.</td></tr>
</table>

rootfstype	The root filesystem type. rootfstype=type حاول وصل لنظام ملفات الجذر بهذا النوع من نظم الملفات. وفي هذه الحالة؛ rootfstype=ext3.
rw	Mount the root device read-write on boot. الوضع الافتراضي للنواة هو عمل وصل لقرص الجذر في نمط القراء فقط read-only أثناء الإقلاع. هذا الخيار يوصل قرص الجذر في نمط القراءة والكتابة read-write عوضا عن ذلك.

Init Options

init هي أولى العمليات التي يجب الابتداء بها من قبل النواة، وهي الجد الأعلى لجميع العمليات الأخرى. هذه الخيارات تتحكم في ماهية البرنامج الذي يعمل وكيفية عمله.

init	Program to run at init time. init=filename شغل ملف ثنائي معين كعملية أم *init* بدلا من البرنامج الافتراضي */sbin/init* .
rdinit	Run the init process from the ramdisk. rdinit=full_path_name شغل برنامج محدد من خلال الاسم والمسار *full_path_name* كعملية أم *init* process . هذا الملف يجب أن يكون على قرص الذاكرة الخاص بالنواة بدلا من وجوده على نظام ملفات الجذر.
S	Run init in single-user mode. الوضع الافتراضي للنواة هو تشغيل *Init* في نمط تعدد المستخدمين multi-user . هذا الخيار يشغل Init في نمط المستخدم الوحيد single-user بدلا من ذلك.

---

kexec Options

النظام الفرعي kexec هو ميزة مخصصة لإعادة التشغيل تعمل على سرعة إعادة التشغيل

وعادة تحتوي على تسهيل ل kdump يتيح تخزين ذاكرة النواة السابقة إلى مكان آمن من أجل تحليلها في وقت لاحق. هذه الخيارات تعدل معاملات النظام الفرعي kexec.

---

crashkernel	Reserve a portion of physical memory for kexec to use. crashkernel=n[KMG]@start[KMG] لنظام الفرعي kexec يميل إلى الحصول على جزء من الذاكرة الفعلية مدخرة للمستقبل. هذا الخيار يدخر الذاكرة لبقية النواة وسوف ينتقل لاستخدامها عند حدوث ارتباكات بالنواة. *n* تحدد مقدار الذاكرة التي يتم توفيرها، و *start* تحدد مكان أجزاء هذه الذاكرة.وكلاهما يقاس بوحدات الكيلوبايت(K)،والميجابايت (M)، أو بالجيجابايت (G).
elfcorehdr	Start of the kernel core image ELF header. elfcorhdr=*n* النواة – مثل جميع البرامج التنفيذية في لينكس- يتم تخزينها بصيغة ELF [1].هذا الخيار يحدد العنوان المكان الفعلي حيث تبدأ منه ترويسة ELF header الخاصة بصورة النواة. وهذا يستخدم من خلال kexec لإيجاد النواة عند إقلاع الصورة الثانوية للنواة.

RCU Options

يعتبر Read Copy Update (RCU) [2]جزءا من النواة يعالج الإقصاء المتبادل لمجموعة متنوعة من النظم الفرعية في شكل عديم القفل lockless . وهناك عدد من الخيارات التي يمكن استخدامها لضبط RCU بطرق مختلفة :

---

(1)، ELF : اختصارا ل **free file format** "صيغة الملفات الحرة"وهي صيغة للملفات تتمتع بكامل المواصفات المتاحة بحرية بلا قيود (مثل القيود القانونية أو التقنية) على استخدامها. ويجوز للمستخدمين تصميم واستخدام الاختلافات التي تناسب احتياجاتهم ، والمساهمة في التحسينات بالجهود الممكنة والموحدة ، ودمجها في الإصدارة الرسمية المقبلة لهذه الصيغة .

(2)RCU : هي تقنية للنواة تعمل على تحسين الأداء على أجهزة الحاسب المحتوي على أكثر من وحدة للمعالجة المركزية. ومن الناحية الفنية هي آلية تزامنية يمكن أحيانا أن تستخدم كبديل لقفل القراء والكتابة readers-writer lock مما يعمل على تقليل الجهد للغاية.

rcu.blimit	RCU batch limit. rcu.blimit=*n* اضبط الحد الأقصى لعدد ردود النداء callbacks لتعمل في دفعة واحدة.
rcu.qhimark	RCU queue high level. rcu.qhimark=*n* يتم وقف حدود الدفعة عندما يكون عدد ردود النداء callbacks المصطفة الخاصة ب RCU تزيد عن *n*.
rcu.qlowmark	RCU queue low level. rcu.qlowmark=*n* يعاد تفعيل حدود الدفعة عندما يكون عدد ردود النداء callbacks المصطفة الخاصة ب RCU أقل من *n*.
rcu.rsinterval	RCU callback queue length. rcu.rsinterval=n ضع عدد ردود النداء الخاصة ب RCU التي ينبغي صفها قبل فرض الجدولة على جميع المعالجات.

ACPI Options

هذه الخيارات تتحكم في المعاملات التي يمكن استخدامها من خلال النظام الفرعي Advanced Configuration and Power Interface (ACPI)

acpi	ACPI subsystem options. acpi=[force|off|noirq|ht|strict] هذا هو الخيار الرئيسي لـ ACPI والقيم هي : *force* تفعيل ACPI بالقوة. ويمكن استخدامه لإبطال خيار تهيئة النواة الذي يوقفه. *off* أوقف عمل ACPI ويمكن استخدامه لإبطال خيار تهيئة النواة الذي يفعله.

	noirq منع ACPI من ان يستخدم من أجل توجيه طلب المقاطعة IRQ. *ht* شغل فقط ما يكفي طبقة ACPI لتفعيل خاصية تعدد خيوط المعالجة HyperThreading على المعالجات القادرة على ذلك. *strict* اجعل طبقة ACPI أقل تساهلا مع المنصات غير المتوافقة تماما مع مواصفات ACPI .
acpi_sleep	ACPI sleep options. acpi_sleep=[s3_bios],[s3_mode] بعد عودة S3 (وذلك حدث بعد ما أصبحت الأجهزة تقوم بعملية النوم في الذاكرة) احتاج العتاد لإعادة تشغيله بشكل سليم .بالنسبة لمعظم الأجهزة فذلك أمر بسيط.فيما عدا بطاقات الفيديو، التي تعمل في العادة من خلال البيوس .وليس لدى النواة المعلومات الكافية لاسترجاع بطاقة الفيديو، لأن المعلومات الخاصة بها موجودة في البيوس وليست قابلة للعمل مطلقا. هذا الخيار يجعل النواة تحاول استخدام نظام ACPI لاستعادة بطاقة الفيديو بطرقتين مختلفتين . انظر الملف *Documentation/power/video.txt* لمزيد من المعلومات عن هذا الخيار؛ ولمعرفة كيفية العثور على القيمة الصحيحة لنوع الجهاز الخاص بك.
acpi_sci	ACPI System Control Interrupt trigger mode. acpi_sci=[level|edge|high|low] ضبط نمط ال trigger mode للتحكم في طلب المقاطعة الخاصة بنظام ACPI.
acpi_irq_balance	Disable ACPI IRQ balance. يجعل ACPI يوازن طلبات المقاطعة النشطة .وهذا هو الخيار الافتراضي عند التشغيل في نمط APIC.
acpi_irq_nobalance	Disable ACPI IRQ balance. يجعل ACPI لا ينتقل إلى طلبات المقاطعة النشطة .وهذا هو الخيار الافتراضي عند التشغيل في نمط PIC.

acpi_irq_isa	Mark the listed IRQs as used by ISA. acpi_irq_isa=irq[,irq...] إذا كان خيار توازن طلب المقاطعة مفعلا ، ضع علامة بالقائمة على المقاطعات المستخدمة في القائمة عن طريق نظام ISA الفرعي.
acpi_irq_pci	Mark the listed IRQs as used by PCI. acpi_irq_pci=irq[,[irq...] إذا كان خيار توازن طلب المقاطعة مفعلا ، ضع علامة بالقائمة على المقاطعات المستخدمة في القائمة عن طريق نظام PCI الفرعي.
acpi_os_name	.ake the operating system name to ACPI acpi_os_name=name أخبر بيوس ال ACPI ان اسم نظام التشغيل العامل هو *name*. وذلك يمكن الاستفادة منه في خداع البيوس لتعتقد أن النظام العامل الآن هو ويندوز بدلا من لينكس. مما يساعد على حل بعض أمور ACPI الخاصة بنسخ البيوس القديمة. على سبيل المثال ؛ استخدم العبارة Microsoft 2001 في خداع البيوس لتعتقد أن ويندوز 2001 هو النظام العامل الآن على الجهاز.
acpi_osi	.Disable the _OSI ACPI method acpi_osi=[*n*] هذا في الواقع خيار مزدوج رغما عن قيمة العدد الصحيح .إذا كانت *n* غائبة، سيقوم ACPI بتعطيل الطريقة _OSI method، وإذا كانت *n* موجودة سيتم تعطيل الطريقة _OSI method.
acpi_serialize	Force serialization of AML methods. فرض تسلسلية طرق لغة الآلة الخاصة ب ACPI.
acpi_skip_timer_override	Skip interrupt override issues. يسمح لطبقة ACPI بتنظيم وتجاهل الأمور الخاصة بتجاوز مقاطعة IRQ0/pin2 لنسخ البيوس التالفة الخاصة بشرائح nForce2 ، التي تحدث في السلوك غير المتوقع لمؤقت PIC[1]-XT .

(1) PIC : Programmable Interrupt Controller متحكم المقاطعة القابل للبرمجة.

acpi_dbg_layer	ACPI debug layer. acpi_dbg_layer=*n* ضبط طبقات تنقيح ACPI debug layers. حيث *n* عدد صحيح يشير بكل جزء إلى طبقة تنقيح مختلفة ل ACPI . بعد تمام إقلاع النظام يمكن ضبط طبقات التنقيح عن طريق الملف */proc/acpi/debug_layer* .
acpi_fake_ecdt	ECDT workaround. يتيح -إذا كان موجودا- ل ACPI اتخاذ طرق جانبية لتفادي إخفاقات البيوس عندما تحتاج إلى جدول مواصفات المتحكم المضمن Embedded Controller Description Table.
acpi_generic_hotkey	Use generic ACPI hotkey driver. إجبار النواة على افتراض أن عداد الدقة pmtimer [2] الخاص بالجهاز يمتلك قيمة ودائما يرجع بقيم جيدة.
ec_intr	ACPI Embedded Controller interrupt mode. ec_intr=n حدد نمط المقاطعة الخاص بالمتحكم المضمن في ACPI.إذا كانت *n* تساوي 0، سيتم استخدام نمط الاقتراع polling mode، وإلا فسوف يتم استخدام نمط المقاطعة Interrupt mode. ونمط المقاطعة هو الافتراضي.
memmap	Mark specific memory as ACPI data. memmap=n[KMG]#start[KMG] يقوم بعمل علامة على مكان ونطاق محدد من الذاكرة كبيانات ACPI. حيث *n* هو حجم نطاق الذاكرة. و *start* هو الموضع الذي يبدأ عنده نطاق الذاكرة، وكلاهما يقاس بوحدات الكيلوبايت(K)، والميجابايت(M)، أو بالجيجابايت(G).
memmap	Mark specific memory as reserved. memmap=n[KMG]$start[KMG] يقوم بعمل علامة على مكان ونطاق محدد من الذاكرة كذخيرة للمستقبل. حيث *n* هو حجم نطاق الذاكرة. و *start* هو الموضع الذي يبدأ عنده نطاق الذاكرة.

Power Management Timer(2)

pnpacpi	Turn Plug and Play ACPI off. pnpacpi=off إيقاف خاصية Plug and Play الخاصة ب ACPI.
processor.max_cstate	Limit the processor to a maximum C-state. processor.max_cstate=*n* تقييد المعالج بحد أقصى ل C-state، ولا عبرة بما تقوله جداول ACPI عما يمكنه دعمه . *n* هو قيمة صالحة ل *C-state* . *والقيمة 9 تلغي أي حد لقائمة الممنوعات الخاصة ب DMI التي ربما تكون موجودة لهذا المعالج .*
processor.nocst	Ignore the _CST method for C-states. يجعل حزمة ACPI تتجاهل طريقة _CST method الخاصة بتحديد المعالج ل C-states واستخدام طريقة FADT الموروثة بدلا من ذلك.

خيارات SCSI

هذه الخيارات تحدد المعاملات المختلفة التي يمكن لنظام SCSI الفرعي استخدامها .وهناك عدد من الخيارات لمشغلات محددة ل SCSI متاحة أيضا .لمزيد من التفاصيل برجاء قراءة ملفات الوثائق الخاصة بالمشغلات المختلفة بداخل دليل النواة *. /Documentation/scsi*

max_luns	Maximum number of SCSI LUNS to probe. max_luns=*n* يعين الحد الأقصى لعدد الوحدات المنطقية LUNs [(1)] SCSI التي يجب على النظام تحقيقها. *n* عدد صحيح من 1 إلى 4,294,967,295.

(1) LUN : Logical_Unit_Number هو مصطلح في مجال وسائط التخزين يعبر عن عدد الوحدات المنطقية ، والوحدة تعبر عن كينونة في بوتوكول سكازي يمكن في كل مرة معالجة عنوان واحد منها من خلال عمليات الدخل والخرج الفعلية input/output (I/O). وكل غرض لسكازي يوفر واحدة أو أكثر من هذه الوحدات المنطقية ولا يعبر عن بيانات الدخل والخرج نفسها ولكنها نيابة عن وحدة منطقية معينة.

max_report_luns	Maximum number of SCSI LUNS received. max_report_luns= *n* يعين الحد الأقصى لعدد SCSI LUNs التي يستطيع النظام استقبالها ، *n* عدد صحيح من 1 إلى 16,384.
scsi_dev_flags	SCSI black/white list. scsi_dev_flags=vendor:model:flags هذا الخيار يسمح للمستخدم بإضافة خانات إلى قائمة black/white الخاصة بسكازي لتحديد بائع وموديل الجهاز.

خيارات PCI

هذه الخيارات توضح معاملات مختلفة التي يمكن لنظام PCI الفرعي استخدامها:

PCI	pci=*option*[,*option*...] كل خيار *option* يمكن أن يكون واحدا مما يلي : *off* لا تقم باختبار ناقل probe. *bios* افرض استخدام PCI BIOS بغير تشغيل العتاد مباشرة.وذلك يعني أنه يجب على النواة الثقة في البيوس .وذلك فعل لأمر غير قياسي (حيث إن برامج البيوس يعرف عنها أنها في الغالب تكذب أكثر مما تصدق). استخدم ذلك فقط غذا كان جهازك PCI host bridge غير قياسي والطريقة العادية للإقلاع لا تعمل معه بشكل سليم. *nobios* لا تستخدم PCI BIOS، ولكن بدلا من ذلكقم بتشغيل العتاد مباشرة. تلك هي الطريقة الافتراضية للتحقق من أجهزة PCI في جميع نسخ النواة بعد 2.6.13 *conf1* افرض استخدام ميكانيكية رقم 1 لتهيئة PCI (وهي طريقة للوصول إلى ذاكرة PCI على أجهزة i386). *conf2* افرض استخدام ميكانيكية رقم 2 لتهيئة PCI (وهي طريقة للوصول إلى ذاكرة PCI على أجهزة i386).

nommconf

عطل استخدام جدول ACPI MMCONFIG الخاص بتهيئة ACPI.

nomsi

إذا كان معامل التهيئة للخيار PCI_MSI مفعلا، يمكن استخدام لخيار إقلاع النواة هذا في تعطيل النطاق النظامي لمقاطعات MSI.

nosort

لا تقم بترتيب أجهزة PCI بناء على الأمر المعطى من قبل بيوس ال PCI. هذا الترتيب يحدث للحصول على أمر للجهاز متوافق مع معظم النسخ الأقدم من النواة.

biosirq

استخدم نداءات بيوس ال PCI للحصول على جدول توجيه المقاطعة interrupt routing table. هذه النداءات معروف بأنها مضرة على العديد من الأجهزة وتسبب تعليقها - hang عند استخدامها.ولكن على أجهزة أخرى تعتبر هي الطريقة الوحيدة للحصول على جدول توجيه المقاطعة.قم بتنفيذ هذا الخيار إذا لم تستطع النواة توزيع المقاطعات أو اكتشاف النواقل الثانوية ل PCI على لوحتك الأم.

Rom

إسناد معالجة حيز العنوان address space لتمديد ذاكرات القراءة فقط ROMs. استخدم هذا الخيار بشيء من الحذر، حيث إن بعض الأجهزة تتشارك مفكات شفرة العناوين – decoders - بين ال ROMs وبين غيرها من الموارد.mask.

irqmask=0xnnnn

ضع قناع البت -bit mask- لطلبات المقاطعة لإتاحة إسنادها تلقائيا لأجهزة PCI. يمكنك أن تجعل النواة تمنع جدول المقاطعات الخاص ببطاقات ISA بهذه الطريقة.

pirqaddr=0xn

حدد العنوان الفيزيائي لجدول المقاطعات -PIRQ table- (والذي يتم توليده عادة من خلال البيوس) غذا كان ذلك خارج نطاق F0000–100000 (سداسي عشري).

lastbus=n

قم بعملية مسح لجميع النواقل خلال الناقل *n*. *وهذا* يمكن

الاستفادة منه عندما لا تتمكن النواة من العثور على النواقل الثانوية ، وتريد إعلام النواة صراحة عن الناقل المراد.

Assign-busses

دائما استخدم أرقام نواقل PCI الخاصة بك، متجاوزا ما يحتمل قيام البرنامج الثابت firmware بفعله.

Usepirqmask

قم بالوفاء قدر الإمكان بقناع المقاطعة IRQ المخزن في جدول BIOS $PIR.وذلك لازم على بعض النظم التي عليها برامج بيوس تالفة، ولا سيما الأجهزة المحمولة من نوع HP Pavilion N5400 و Omnibook XE3 ، وهذا لن يكون له تأثير في حالة تفعيل ACPI IRQ routing .

Noacpi

لا تستخدم ACPI من أجل توجيه المقاطعة أو فحص PCI.

Routeirq

قم بتوجيه المقاطعة لجميع أجهزة .وذلك يتم فعله بشكل عادي في ( ) pci_enable_device ، حيث إن هذا الخيار يعمل بشكل مؤقت في الالتفاف حول المشغلات التالفة التي لا تستطيع استدعاءها.

firmware

لا تقم بإعادة سرد الناقل ،ولكن بدلا من ذلك قم فقط باستخدام الإعدادات من محمل الإقلاع bootloader.وهذا يستخدم حاليا في نظام IXP2000 حيث يجب تهيئة الناقل بطريقة معينة للمعالجات الملحقة .

Plug and Play BIOS Options

Disable the ISA Plug and Play (PnP) subsystem. عطل النظام الفرعي ISA Plug and Play ، إذا كان مفعّلا في تهيئة النواة؟	noisapnp

pnpbios	PnP BIOS settings. pnpbios=[on|off|curr|no-curr] اضبط الإعدادات الرئيسية ل PnP BIOS. حيث on تفعّل النظام الفرعي PnP BIOS ، و off توقفه ، و curr تبلغ النظام الفرعي PnP BIOS بأن يستخدم الإعدادات الثابتة ، و no-curr تبلغ النظام الفرعي للتحقق من الإعدادات الديناميكية قدر الإمكان.
pnp_reserve_irq	PnP BIOS reserved IRQs. pnp_reserve_irq=irq1[,irq2...] ضع قائمة بالمقاطعات التي ينبغي ألا يستخدمها نظام PnP BIOS للتهيئة التلقائية.
pnp_reserve_dma	PnP BIOS reserved DMAs. pnp_reserve_dma=dma1[,dma2...] ضع قائمة بال DMAs التي ينبغي ألا يستخدمها نظام PnP BIOS للتهيئة التلقائية.
pnp_reserve_io	PnP BIOS reserved I/O ports. pnp_reserve_io=io1,size1[,io2,size2...] منافذ الدخل والخرج التي ينبغي ألا يستخدمها نظام PnP BIOS للتهيئة التلقائية. كل منفذ يدرج بالقائمة بناءا على مكان ابتدائه وحجمه .
pnp_reserve_mem	PnP BIOS reserved memory regions. pnp_reserve_mem=mem1,size1[,mem2,size2...] نطاقات الذاكرة التي ينبغي ألا يستخدمها نظام PnP BIOS للتهيئة التلقائية. كل منفذ يدرج بالقائمة بناءا على مكان ابتدائه وحجمه .

SELinux Options

هذه الخيارات تغير بعض الجوانب الأساسية لبدء تشغيل SELinux .

Set the initial checkreqprot flag value. checkreqprot=[0|1] ضع قيمة ل initial checkreqprot flag 0 تعني أن فحص الحماية سيتم تفعيله من خلال النواة إضافة إلى تنفيذ أي حماية ضمنية. 1 تعني أن طلب فحص الحماية وقع من أحد البرامج. القيمة الافتراضية موضوعة من خلال خيار تهيئة النواة .	checkreqprot
Set the initial enforcing status. enforcing=[0|1] تحديد ماهية القواعد التي سيفرضها SELinux عند الإقلاع . القيمة 0 تعني أن SELinux سوف يسجل فقط سياسة الانتهاكات policy violations، لكنه لن يمنع الوصول الى أي شيء . القيمة 1 تعني أن وضع التنفيذ سيكون مفعلا تماما مع الموانع بالإضافة إلى التسجيل أيضا . القيمة الافتراضية هي 0. يمكن تغيير هذه القيمة في وقت التشغيل عن طريق ملف *. /selinux/enforce*	enforcing
Enable or disable SELinux at boot time. selinux=[0|1] هذا الخيار يتيح تفعيل (1) SELinux أو إيقافه (0) وقت الإقلاع . القيمة الافتراضية توضع من خلال خيار تهيئة النواة ، إذا تم تفعيل SELinux أثناء الإقلاع ، ويمكن استخدام الملف */selinux/disable* فيما بعد لإبطاله قبل تحميل السياسة الأولية initial policy.	selinux
Set the network control model. selinux_compat_net=[0|1] وضع القيمة الأولية لموديل التحكم في الشبكة ل SELinux . القيمة 0 تستخدم مجموعة ضوابط secmark-based الحديثة. والقيمة 1 تستخدم مجموعة الضوابط القديمة. 0 هي القيمة الافتراضية والمفضلة. يمكن تغيير هذه القيمة في وقت التشغيل عن طريق الملف */selinux/compat_net/* .	selinux_ compat_net

خيارات الشبكة

هذه الخيارات تتحكم بالأمور منخفضة المستوى لنظام الشبكات.

netdev	Set various network device parameters. netdev=[irq],[io],[mem_start],[mem_end],[name] تحديد معاملات بطاقة الشبكة ، التي تحدد المشغل المستخدم من قبل بطاقة الشبكة.بعض الملفات المصدرية الخاصة بالمشغل توثق الخيارات القابلة للتطبيق. هذه الخيارات لا تطبق في العادة على أجهزة PCI، أو USB، أو غيرها من أجهزة الشبكة من نوع plug-and-play . وذلك بغرض استخدامه فقط على الأجهزة التي لا يمكنه اكتشاف المهام الخاصة بها.
rhash_entries	Set the number of route cache hash buckets. dhash_entries=*n* هذا الخيار يسمح لك بإلغاء العدد الافتراضي لل hash buckets الخاص بال route cache في النواة . وينصح به فقط لخبراء الشبكات.
shapers	Set the maximum number of network shapers. shapers=*n* هذا الخيار يتيح لك تحديد الحد الأقصى لعدد network shapers التي يمكن للنواة استخدامها.
thash_entries	Set the number of TCP connection hash buckets. thash_entries=*n* هذا الخيار يتيح لك تجاوز العدد الافتراضي من hash buckets ل TCP connection cache الخاص بالنواة.

خيارات نظام ملفات الشبكة NFS

هذه الخيارات تتحكم في بدء تشغيل NFS .

lockd.nlm_grace_period	Assign a grace period to the lock manager. lockd.nlm_grace_period=n ضبط المهلة الزمنية لمدير قفل NFS. وتقاس *n* بالثانية.

lockd.nlm_tcpport	.Assign a TCP port to the lock manager `lockd.nlm_tcpport=port` تحديد منفذ TCP الذي ينبغي على NFS lock manager أن يستخدمه. *port* يجب أن تكون قيمة صالحة لمنفذ TCP .
lockd.nlm_timeout	Assign a new timeout value to the lock manager. lockd.nlm_timeout=*n* تجاوز القيمة الافتراضية ل NFS lock manager.وتقاس *n* بالثانية.إذا لم يتم تحديد هذا الخيار، ستكون القيمة الافتراضية المستخدمة هي 10 ثوان .
lockd.nlm_udpport	Assign a UDP port to the lock manager. lockd.nlm_udpport=*port* ضبط منفذ UDP الذي ينبغي على NFS lock manager أن يستخدمه. *port* يجب أن تكون قيمة منفذ صالحة ل UDP.
nfsroot	Specifies the NFS root filesystem. nfsroot=[server-ip:]root-dir[,nfs-options] حدد نظام الملفات الجذر ل NFS على الأجهزة عديمة القرص، للسماح لها بالإقلاع من خلال NFS بشكل سليم. إذا لم يتم وضع هذا المعامل ، سيتم استخدام القيمة */tftp-boot/client_ip_address* كنظام ملفات الجذر مع الخيارات الافتراضية ل NFS . *Server-ip* عنوان ip الذي يتصل به خادم NFS. *Root-dir* الدليل الذي على خادم NFS وصله كنظام ملفات جذر.إذا كانت هناك العلامة %s في هذه العبارة، فسيتم استبدالها بتمثيل ASCII لعنوان Ip الخاص بالعميل. *Nfs-options* خيارات NFS القياسية مثل ro, المتبوعة بفاصلة.

Set the NFSv4 TCP port for the callback channel. nfs.callback_tcpport=port حدد منفذ TCP الذي ينبغي الإصغاء له من خلال قناة رد النداء الخاصة ب NFSv4. و *port* يجب أن تكون قيمة صالحة لمنفذ TCP.	nfs.callback_ tcpport
Set the maximum lifetime for idmapper cache entries. nfs.idmap_cache_timeout=*n* وضع الحد الأقصى لعمر خانات تخزين idmapper. وتقاس *n* بالثواني .	nfs.idmap_ cache_timeout

Hardware-Specific Options

هذه الخيارات مخصصة لمعاملات مختلفة بناء على العتاد الموجود على النظام.

Disable the USB subsystem. إذا كان هذا الخيار موجودا ، فلن يتم تشغيل نظام فرعي USB.	nousb
Parallel port and its mode. تحديد المنفذ المتوازي الذي سيستخدم. الصيغة lp=port1,port2... تشير إلى سلسلة من المنافذ المتوازية للأجهزة ، تبدا بـ lp0.وكمثال lp=none,parport0 الذي يبطل إعدادات الجهاز lp0، ويجعل الجهاز lp1 يستخدم المنفذ المتوازي الأول. Lp=0 يعطل مشغل الطابعة. lp=reset يؤدي إلى إعادة تشغيل الطابعات الملحقة . هذا الخيار يمكن أن يحتوي على مواصفات المنفذ . lp=auto يجعل النواة تفحص رقم هوية - ID - الجهاز في كل منفذ للتحقق من ماهية الطابعة المتوافقة مع IEEE 1284- compatible والموصلة بالجهاز . فإذا تم ذلك فستقوم النواة بتشغيل هذه الطابعة.	lp

parport	Specify the parallel port parameters. parport=[setting[,setting...] يحدد مشغلات المنفذ المتوازي . المنافذ المتوازية يتم إسناد المهام المخصصة لها عن طريق سطر الأوامر، بداية ب parport0. إرغام المشغل تلقائيا على استخدام اي إعدادات مضبوطة من نوع IRQ/DMA (الوضع الافتراضي هو تجاهل الضبط التلقائي لإعدادات IRQ/DMA بسبب إمكانية وقوع تعارض بينها). يمكنك أيضا تحديد العنوان الأساسي، والمقاطعة، وإعدادات DMA، بصيغة 0xnnnn[,irq[,dma]] ، ويمكن أن تكون irq و dma أرقاما، أو استخدام إعدادات مضبوطة تلقائيا لهذا المنفذ المخصص ، أو nofifo لتجنب استخدام FIFO حتى ولو تم التعرف عليه .
parport_init_mode	Parallel port initialization mode. parport_init_mode=[spp|ps2|epp|ecp|ecpepp] يحدد نمط تشغيل المنفذ المتوازي. وهذا أمر ضروري على حاسبات Pegasos حيث إن ال firmware لا يحتوي على خيارات لإعداد نمط عمل المنفذ المتوازي . هذا الخيار يعمل على رقائق المنفذ المتوازي من نوع 686a و 8231 .
nr_uarts	Maximum number of UARTs to be registered. nr_uarts=n يعين الحد الأقصى لعدد UARTs المختلفة التي يمكن تسجيلها بداخل النواة .

Timer-Specific Options

هذه الخيارات تبطل عمل السلوك الافتراضي للنواة في إصلاح المشكلات الخاصة بشرائح معينة.

enable_timer_pin_1	Enable pin 1 of the APIC timer. تفعيل pin 1 من مؤقت APIC. هذا الخيار يمكن أن يكون مفيدا للالتفاف حول bugs الرقاقات (على بعض رقاقات ATI على وجه الخصوص). تحاول النواة وضع افتراض معقول، ولكن أحيانا يكون هذا الخيار ضروريا لتخطي ذلك الأمر.

Disable pin 1 of the APIC timer. تفعيل pin 1 من مؤقت APIC. هذا الخيار يمكن أن يكون مفيدا لنفس الأسباب المذكورة في enable_timer_pin_1 .	disable_timer_pin_1
Enable interrupt 0 timer routing over the 8254 chip. قم بتفعيل interrupt 0 timer routing عبر رقاقات 8254 إضافة للتوجيه عبر IO-APIC. تحاول النواة وضع افتراض معقول ، ولكن أحيانا يكون هذا الخيار ضروريا لتخطي ذلك.	enable_8254_timer
Disable interrupt 0 timer routing over the 8254 chip. قم بتعطيل interrupt 0 timer routing عبر رقاقات 8254 إضافة للتوجيه عبر IO-APIC. تحاول النواة وضع افتراض معقول ، ولكن أحيانا يكون هذا الخيار ضروريا لتخطي ذلك.	disable_8254_timer
Disable HPET and use PIT instead. hpet=disable قم بتعطيل مصدر المؤقت HPET وأبلغ النواة باستخدام مصدر المؤقت PIT عوضا عن ذلك.	hpet
Set the specific clocksource. clocksource=[hpet|pit|tsc|acpi_pm|cyclone|scx200_hrt] إلغاء ال clocksource الافتراضي للنواة واستخدام clocksource باسم معين بدلا من ذلك.	clocksource

Miscellaneous Options

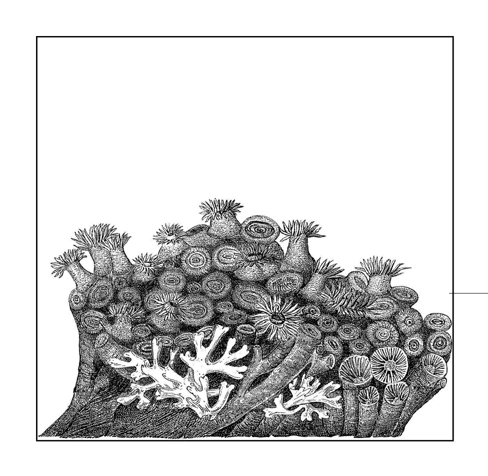
هذه الخيارات يجب أن تكون متاحة دائما، ولا يتوقف ذلك على أي نظام فرعي أو عتاد محدد يكون موجودا على النظام كي يعمل بشكل سليم.

.Set the number of dentry hash buckets dhash_entries=n هذا الخيار يسمح لك بتجاوز العدد الافتراضي لحاويات الدفعة ل kernel's dentry cache وينصح به لخبراء النواة فقط.	dhash_entries

elevator	Set the default I/O scheduler elevator. elevator=[anticipatory|cfq|deadline|noop] يحدد جدول الدخل/الخرج. انظر الفصل 11 لرؤية قائمة مجدولات الدخل/الخرج المتاحة، ووظيفة كل منها.
hashdist	Distribute large hashes across NUMA nodes. hashdist=[0|1] الدفعات الكبيرة التي توجد خلال عملية الإقلاع على منصات IA-64 ، تكون افتراضيا موزعة عبر عقد NUMA المختلفة.هذا الخيار يسمح للمستخدم بتشغيل أو إيقاف on/off هذا الخيار.
combined_ mode	Specify IDE driver usage. combined_mode=[combined|ide|libata] التحكم في ماهية المشغل الذي سيستخدم منافذ IDE في النمط المشترك : مشغلات IDE التقليدية، و libata أو كلاهما . علما بأن هذا الاستخدام لخيار IDE ، أو libata ربما يؤثر على تسمية الأقراص (فمثلا يغير hdc إلى sdb).
max_loop	.Maximum number of loopback devices max_loop=n تحديد الحد الأقصى لعدد أجهزة نظم ملفات loopback التي يمكن وصلها في نفس الوقت. n عدد صحيح من 1 إلى 256.
panic	Time to wait after panic before rebooting. panic= n تحديد المقدار الزمني بالثواني والذي ينبغي على النواة انتظاره بعد حدوث ارتباك panic قبل أن تعيد التشغيل. إذا تم ضبطه على القيمة 0 (وهو القيمة الافتراضية) فلن تقوم النواة بإعادة الإقلاع بعد عملية الارتباك panicking؛ ولكن ببساطة سوف تنطفئ.

pause_on_oops	Delay between kernel oopses. pause_on_oops=*n* إبلاغ النواة بأن تطفئ جميع المعالجات بعد أول oops[1] لمدة *n* من الثواني قبل مواصلة العمل . وذلك مفيد إذا كانت رسائل oopses ما زالت تنسدل على الشاشة قبل أن تستطيع تسجيل ما عليها أو أخذ صورة لها .
profile	Control the kernel profiling. profile=[*schedule,*][*number*] هذا الخيار يؤثر على كيفية حساب profiler النواة . إذا تم تحديد الجدول ، سيتم تأثر عناصر الجدول بالقيم الموضوعة في *number*. إذا لم يتم تحديد قيمة ل *schedule*، يكون *number* هو حجم الخطوة بمقدار power of two لإحصائيات ال profiling الزمني في النواة. الاستخدام الشائع لهذا الخيار هو profile=2.

(1) Oops : كلمة تعني حرفيا "ويحي ، عفوا، معذرة"وتقال عند خطأ السهو أو غير المتوقع وهو من أسماء الأصوات .وفي لينكس هو رسالة عن انحراف عن السلوك الصحيح من نواة لينكس والتي تنتج سجلا لخطأ معين . والمثال الأكثر شهرة لهذا المعنى ما يعرف بحالة الذعر، أو ارتباك النواة kernel panic والناتج بسبب أنواع كثيرة من Oops ، لكن قد يسمح لعملية أخرى بمواصلة العمل مع شيء من الاشتباه وضعف الموثوقية . وعند اكتشاف النواة لأي مشكلة تقوم بطباعة رسالة Oops وتقوم بقتل أي عملية مضرة ، وهذه الرسالة يستخدمها مهندسو النواة في تصحيح هذا الوضع التي خلقت هذا ال Oops وإصلاح الخطأ البرمجي الذي يسبب ذلك.



10

مرجع بأوامر بناء النواة

كما ناقشنا في الفصل الرابع ، الأداة التي تربط
أجزاء النواة هي برنامج make ، الذي يتم تمريره للأغراض targets المحددة التي
تريد بناءها.
الفصل الرابع يذهب أبعد من الأوامر الأساسية اللازمة لبناء النواة على النحو
الصحيح ، ولكن
بناء نواة النظام يتضمن أيضا مجموعة واسعة من الأوامر الأخرى. هذا الفصل يوضح
تفاصيل
عن هذه الأوامر ، وما يمكن أن تستخدم من أجله.
كل هذه الأوامر تمرر إلى البرنامج make على سطر الأوامر ، وعدد منها ، يمكن
تجميعه معا إذا رغبت في ذلك . على سبيل المثال :

```
$ make mrproper xconfig
```

هذه الأوامر تتكسر إلى أنواع مختلفة كما يتضح في الأجزاء التالية .ويمكنك الحصول على
ملخص لمعظم هذه الأوامر عن طريق كتابة الأمر وأنت داخل دليل بناء النواة :

```
$ make help
```

هذه الأوامر تطبع العديد من أهداف برنامج make الشائعة والتي سيتم شرحها في بقية هذا
الفصل.

الأغراض الإعلامية Informational Targets:

جدول 10-1 يعرض الأوامر التي تطبع رقم إصدار النواة،على أساس رقم الخيارات
المختلفة. وهي تسخدم عادة من قبل السكربتات لتحديد نسخة النواة التي سيتم
بناؤها .

جدول 10-1: Informational targets

الأمر	الوصف
Kernelrelease	يعرض نسخة النواة الحالية والتي تحددت من قبل برنامج البناء.
Kernelversion	يعرض رقم نسخة النواة الحالية التي أبلغ بها من Makefile الرئيسي. وهذا يختلف عن الغرض kernelrelease في أنه لا يستخدم

أي معلومات إضافية عن النسخة على أساس خيارات التهيئة أو ملفات .localversion

Cleaning Targets

جدول 2-10 يوضح الأوامر التي تقوم ببساطة بإزالة ملفات عمليات البناء السابقة. وهذا الاستخدام يوصى به بشدة للتأكد من عدم إفساد الأبنية الجديدة بملفات متبقية مبنية بخيارات مختلفة.وهي تختلف في المدى الذي تزيله ففي بعض الأحيان تريد الاحتفاظ بالملفات التي قمت بإجراء تغييرات عليها.

جدول Cleaning targets:2-10

الغرض	الوصف
clean	إزالة الملفات المتولدة عن نظام بناء النواة مع الاحتفاظ بملفات تهيئة النواة
mproper	إزالة جميع الملفات المتولدة عن نظام بناء النواةبالإضافة إلى ملفات التهيئة وبعض ملفات النسخ الاحتياطي السابقة
distclean	تفعل كل ما يفعله mproper مع إزالة بعض ملفات النسخ الاحتياطي والباتش المتبقية.

Configuration Targets

الجدول 3-10 يوضح الأوامر التي تتيح تهيئة النواة من خلال نطاق واسع من الطرق المختلفة.

جدول Configuration targets 3-10

الأمر	الوصف
config	تحديث ملف تهيئة النواة الحالية باستخدام برنامج سطر الأوامر
menuconfig	تحديث ملف تهيئة النواة الحالية باستخدام برنامج مبني على قائمة نصية

Xconfig	تحديث ملف تهيئة النواة الحالية باستخدام برنامج رسومي مبني على QT
gconfig	تحديث ملف تهيئة النواة الحالية باستخدام برنامج رسومي مبني على GTK+
oldconfig	تحديث ملف تهيئة النواة الحالية باستخدام ملف config. الحالي والتنبيه عند كل الخيارات الجديدة التي اضيفت إلى النواة.
silentoldconfig	يشبه oldconfig، ولكنه لا يطبع أي شيء على الشاشة إلا عند الحاجة لإجابة المستخدم عن أحد الأسئلة.
randconfig	يولد تهيئة جديدة للنواة مع إجابات عشوائية لجميع الخيارات المختلفة.
defconfig	يولد تهيئة جديدة للنواة مع إجابة افتراضية لكل الخيارات.القيم الافتراضية مأخوذة من ملف موجود بالمسار /arch/$ARCH defconfig حيث إن $ARCH تشير إلى معمارية معينة للمعالج الذي تستخدمه النواة للبناء.
allmodconfig	يولد تهيئة جديدة للنواة مع تفعيل لكافة modules وقتما كان ذلك ممكنا.
allyesconfig	يولد تهيئة جديدة للنواة مع ضبط كل الخيارات على yes.
allnoconfig	يولد تهيئة جديدة للنواة مع ضبط كل الخيارات على no.

لاحظ أن الأغراض allyesconfig و allmodconfig و allnoconfig و randconfig
أيضا تستفيد من ميزة متغير البيئة KCONFIG_ALLCONFIG.
فإذا أشار المتغير إلى أحد الملفات سيستخدم هذا الملف كقائمة لقيم التهيئة التي تريد وضعها لقيمة معينة.وبعبارة أخرى فإن الملف يبطل الاستخدامات العادية لبرنامج *make*.
على سبيل المثال؛ إذا كان الملف ~/linux/must_be_set يحتوي على المتغيرات التالية :

```
$ cat ~/linux/must_be_set
CONFIG_SWAP=y
CONFIG_DEBUG_FS=y
```

ثم أدخلت make allnoconfig مع القيمة المناسبة لمتغير البيئة KCONFIG_ALLCONFIG في العمل :

```
$ KCONFIG_ALLCONFIG=../must_be_set make allnoconfig
$ grep CONFIG_SWAP .config
CONFIG_SWAP=y
```

وبعد ذلك تشتمل النتائج على :

```
$ grep CONFIG_DEBUG_FS .config
CONFIG_DEBUG_FS=y
```

هذا المتغير لن يكون في العادة موضوعا خلاف y .

إذا لم يكن المتغير KCONFIG_ALLCONFIG موضوعا،يقوم نظام البناء بالتحقق من الملفات في المستوى الأعلى من مجلد البناء المسماة :

- allmod.config
- allno.config
- allrandom.config
- allyes.config

إذا كان أي من هذه الملفات موجودا يقوم نظام البناء باستخدامهم كقائمة لقيم التهيئة والتي يجب فرض قيم محددة عليها.فإذا لم يوجد أحد هذه الملفات يقوم نظام البناء في النهاية بالبحث عن ملف يدعى *all.config* لفرض قائمة اضطرارية من قيم التهيئة.

يمكنك استخدام هذه الملفات المختلفة لإنشاء قاعدة جيدة ومعروفة للتهيئة والتي سوف تعمل دوما. وبعد ذلك يمكن استخدام خيارات التهيئة الأخرى لإنتاج تهييئات اختبارية مختلفة للأحوال المطلوبة.

Build Targets

الجدول 4-10 يعرض أغراض البناء للنواة نفسها بطرق متنوعة.

جدول 4-10: *Build targets*

الأمر	الوصف
all	يبني جميع الأغراض المختلفة اللازمة لهذه النواة لتكون قابلة للاستخدام.بالإضافة إلى كل من modules و الجزء الثابت من النواة.

vmliux	يبني فقط الجزء الثابت من النواة وليس أي modules قابلة للتحميل.
modules	يبني كل ال modules القابلة للتحميل على النواة لهذه التهيئة.
modules_install	تثبيت كل ال modules في مكان معين.فإذا لم يتم تحديد مكان مع متغير البيئة INSTALL_MODULES_PATH، سيتم التثبيت في الدليل الافتراضي للجذر على الجهاز.
dir/	يبني كل الملفات الموجودة في مجلد محدد وكل المجلدات الفرعية المضمنة تحته.
dir/file.[o|i|s]	يبني فقط الملف الذي يتم تحديده.
dir/file.ko	يبني كل الملفات اللازمة ويربطها سويا لتكوين موديل محدد .
tags	يبني كل الوسوم tags المطلوبة والتي يمكن لمعظم محررات النصوص الشائعة استخدامها أثناء تحرير الشفرة المصدرية.
TAGS	يبني كل الوسوم tags المطلوبة والتي يمكن لمعظم محررات النصوص الشائعة استخدامها أثناء تحرير الشفرة المصدرية.
cscope	يبني صورة ل cscope، وهي مفيدة في عمليات البحث داخل شجرة الملف المصدري،ولشجرة المصدر لمعمارية معينة لملف التهيئة (وليس كل الملفات المصدرية للنواة).

يمكنك أيضا تمرير عدد من متغيرات البيئة لبرنامج make يغير من شكل البناء.وذلك يمكن تحديده لكافة الأغراض،كما هو موضح في جدول 5-10

جدول 5-10:متغيرات البيئة

المتغير	القيمة	الوصف
v	0	يقوم بإبلاغ نظام البناء بالعمل بشكل هادئ، مظهرا فقط الملف الذي يتم بناؤه حاليا،وليس كل الأمر الجاري عمله لبناء هذا الملف.هذا هو الخيار الافتراضي لنظام البناء.
v	1	يقوم بإبلاغ نظام البناء للعمل بطريقة النمط الحواري verbose ، مظهرا كل أحداث الأمر المستخدم لتوليد كل الملفات المحددة.

0	dir	ذلك يبلغ نظام البناء لعرض كل الملفات الناتجة في المجلد dir بالإضافة إلى ملفات تهيئة النواة. وذلك يتيح بناء النواة من نظام ملفات للقراءة فقط read-only (مثل السيديروم) والحصول على الناتج في مكان آخر.
c	1	وذلك يقوم بفحص كافة ملفات C التي سيتم بناؤها مع الأداة *sparse*،والتي تشير إلى الأخطاءالبرمجية الشائعة في الملفات المصدرية للنواة.ويمكن تحميل sparse باستخدام الأمر git من الموقع *git://git.kernel.org/pub/scm/devel/sparse/sparse.git.* ويوجد يوميا نسخ منه على الموقع *http://www.codemonkey.org.uk/projects/git-snapshots/sparse/* والمزيد من المعلومات عن كيفية استخدام sparse يمكنك العثور عليها في ملف *Documentation/sparse.txt* في *شجرة الملف المصدري للنواة.*
c	2	وذلك يقوم بالفحص الجبري لملفات C عن طريق الأداة sparse حتى ولو لم يكن هناك حاجة لبنائها.

Packaging Targets

تقوم هذه الأغراض بتجميع النواة المبنية داخل حزمة وحيدة قائمة بذاتها يمكن تثبيتها على قطاع عريض من الأجهزة المختلفة،كما هو موضح في جدول 6-10

جدول 6-10:Packaging Targets

الأمر	الوصف
rpm	يقوم ببناء النواة أولا ثم يقوم بتحزيمها في حزمة RPM يمكن تثبيتها.
rpm-pkg	يقوم ببناء الحزمة المصدرية RPM التي تحتوي على أساس النواة.
binrpm-pkg	يقوم ببناء حزمة RPM تحتوي على نواة وموديلات مترجمة (compiled).
deb-pkg	يقوم ببناء حزمة ديبيان تحتوي على نواة وموديلات مصنفة.

tar-pkg	تقوم ببناء أرشيف tarball تحتوي على نواة وموديلات مصنفة.
targz-pkg	تقوم ببناء أرشيف مضغوط من نوع gzip tarball تحتوي على نواة وموديلات مصنفة.
tarbz2-pkg	تقوم ببناء أرشيف مضغوط من نوع bzip2 tarball تحتوي على نواة وموديلات مصنفة.

Documentation Targets

جدول 7-10 يوضح الأوامر التي تقوم ببناء الوثائق الداخلية للنواة بمختلف الصيغ المتنوعة.

الأمر	الوصف
xmldocs	يبني وثائق النواة بصيغة ملفات XML DocBook .
psdocs	يبني وثائق النواة بصيغة ملفات PostScript
pdfdocs	يبني وثائق النواة بصيغة ملفات PDF
htmldocs	يبني وثائق النواة بصيغة ملفات HTML
mandocs	يبني وثائق النواة بصيغة صفحات المساعدة manpages، والتي يمكن تثبيتها بعد ذلك من خلال الأمر installmandocs.

Architecture-Specific Targets

كل معمارية نواة تحتوي على مجموعة من الأوامر المحددة الفريدة من نوعها.

ويبين الجدول 8-10 الأغراض المتاحة لمعمارية إنتل 32 بت .

الأمر	الوصف
bzimage	ينشئ صورة مضغوطة للنواة ويضعها في الملف *arch/i386/boot/bzImage*. *وهذا هو الغرض الافتراضي لبناء نواة i386*

install	يقوم بتثبيت صورة النواة باستخدام برنامج معين للتوزيعة *sbin/installkernel/ لاحظ أنه لا يقوم بتثبيت موديلات النواة،ولكن يجب عمل ذلك من خلال الأمر modules_install*
bzdisk	ينشئ صورة إقلاع للقرص المرن ويقوم بكتابتها على الجهاز dev/fd0
fdimage	ينشئ صورة إقلاع للقرص المرن ويقوم بوضعها في الملف *arch/i386/boot/fdImage*
isoimage	ينشئ صورة إقلاع قرص مدمج ويضعها في الملف *arch/i386/boot/Image.iso ويجب وجود الحزمة syslinux على نظامك لتعمل تلك الصورة بشكل سليم.*

Analysis Targets

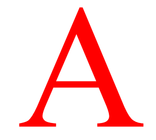
الجدول 10-9 يوضح الأوامر التي يفضل تنفيذها للعثور على أي مشاكل في شفرة النواة.وإنها لفكرة جيدة أن تنشئ قائمة stack space عند إنشاء شيفرة جديدة لتحدد أن تغييراتك لا تشغل حيزا كبيرا من مساحة تكديس- stack space- النواة.

ويعتبر الأمر namespacecheck مفيدا لتحديد ما هي تغييراتك التي يمكنها بأمان إضافة رموزها إلى المجال الاسمي العمومي-[(1)] global namespace الخاص بالنواة.

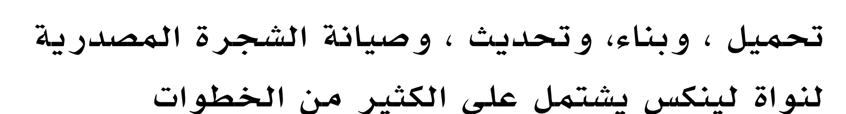
جدول 10-9 Analysis targets

الأمر	الوصف
checkstack	يولد قائمة من الدوال التي تستخدم في أغلب مساحة تكديس stack space للنواة.
namespace check	يولد قائمة من جميع رموز النواة والمجالات الاسمية namespace لها.وهذه ستكون قائمة كبيرة.

(1)Namespace:يترجم بالحيز الاسمي أو المجال الاسمي :وهو عبارة عن وحدة حاوية أو بيئة تحتوي على مجموعة من الأسماء الفريدة من المعرفات والدوال والكلاسات والمتغيرات والثوابت الخاصة بها وهي تتيح للمبرمج تقسيم برنامجه إلى وحدات كل وحدة تسمى namespace ويمكنك تخزين عنصر كمتغير أو كلاس ونحوه بنفس الاسم في أكثر من وحدة namespace ويستطيع الكومبايلر التفرقة بينهما بناء على اسم ال namespace الخاص بكل منهما على سبيل المثال لو أن هناك موظف يحمل id رقم 123 في الشركة X وهناك موظف آخر يحمل نفس ال id في الشركة Y فلا يعتبر ذلك تعارضا حيث يمكن التفرقة بينهما عن طريق الاختلاف في اسم الشركة(namespace).... (المترجم)

ملحق 

أدوات مساعدة



تحميل ، وبناء، وتحديث ، وصيانة الشجرة المصدرية لنواة لينكس يشتمل على الكثير من الخطوات المختلفة ، كما يبين هذا الكتاب. وكما جرت العادة بالنسبة للأشخاص الكسولين ، فقد قام المطورون بإنشاء برامج للمساعدة على القيام بالمهام الروتينية المختلفة.

نحن هنا نشرح عددا قليلا من هذه الأدوات المفيدة والأساسيات المتعلقة بكيفية استخدامها .

تطوير نواة لينكس يختلف من عدة أوجه عن تطوير البرمجيات التقليدية . فهناك بعض المتطلبات الخاصة لمبرمجي النواة تتضمن ما يلي :

- قم باستمرار بتطبيق التغييرات التي قمت بها لمواكبة التطور السريع في جدول الأعمال الخاص بتطوير إصدارات النواة .
- قم بإزالة أي تعارضات بين التعديلات التي قمت بها والتي قام بها أناس آخرون.
- قم بتصدير تعديلاتك في صيغة تتيح للآخرين المشاركة والعمل بها بسهولة.

patch and diff

هذا الجزء قائم على أساس أحد المقالات التي تم نشرها على موقع *Linux Journal*.

أحد أكثر الطرق شيوعا لإنشاء نواة عاملة هو عن طريق استخدام برنامجي *patch* و

diff . لاستخدام تلك الأدوات ، يجب استخدام سلسلتين من المجلدات : أحدهما نظيفة"clean" والأخرى عاملة"working". السلسلة النظيفة هي إحدى إصدارات النواة، بينما العاملة تقوم على النسخة ذاتها، ولكنها تشتمل على التعديلات الخاصة بك . لذلك يمكنك استخدام *patch* و *diff* في استخلاص التعديلات الخاصة بك، وحملها بعد ذلك لإصدارة النواة الجديدة.

على سبيل المثال ، أنشئ مجلدين يحتوي كل منهما على أخر إصدار من النواة، كما تم شرحه في الفصل الثالث :

```
$ tar -zxf linux-2.6.19.tar.gz
$ mv linux-2.6.19 linux-2.6.19-dirty
$ tar -zxf linux-2.6.19.tar.gz
$ ls
linux-2.6.19/
linux-2.6.19-dirty/
```

والآن قم بالتعديلات المختلفة التي ترغب فيها على المجلد *dirty*- واترك مجلد النواة الأصلي الأخر كما هو. بعد أنتائك من عمل التغييرات، ينبغي عليك أن تنشئ باتش لإرساله إلى الأشخاص الآخرين:

```
$ diff -Naur -X linux-2.6.19/Documentation/dontdiff
linux-2.6.19/ \
linux-2.6.19-dirty/ > my_patch
```

وهذا سينشئ ملفا باسم *my_patch* يحتوي على الاختلافات بين ما قمت بعمله وبين مجلد النواة الخالية من التغييرات kernel 2.6.19.

النسخ الجديدة للنواة

إذا تم إطلاق نسخة حديثة من النواة ، ورغبت في نقل تعديلاتك إلى هذه النسخة الجديدة، فسوف تحتاج إلى محاولة تطبيق الباتش الذي قمت بتوليده على نسخة نواة نظيفة. ويمكن عمل ذلك عن طريق الخطوات التالية :

1. إنتاج الباتش الأصلي الخاص بك ، كما في المثال السابق.
2. استخدام الباتش الرسمي من موقع *kernel.org*، وترقية النواة القديمة إلى إصدار أحدث :

```
$ cd linux-2.6.19
$ patch -p1 < ../patch-2.6.20
$ cd ..
```

```
$ mv linux-2.6.19 linux-2.6.20
```

3. قم بنقل الدليل الذي تعمل عليه إلى نسخة أعلى من خلال إزالة الباتش الخاص بك، وبعد ذلك تقوم بتطبيق التحديث الجديد :

```
$ cd linux-2.6.19-dirty
$ patch -p1 -R < ../my_patch
$ patch -p1 < ../patch-2.6.20
$ cd ..
$ mv linux-2.4.19-dirty linux-2.6.20-dirty
```

4. قم بتطبيق الباتش الخاص بك في مقدمة التحديث الجديد :

```
$ cd linux-2.6.20-dirty
$ patch -p1 < ../my_patch
```

إذا لم يتم تنفيذ الباتش الخاص بك بطريق سليمة ، قم بإزالة كل التعارضات التي نشأت (سيقوم سطر أوامر الباتش بإبلاغك بهذه التعارضات، والملفات *.rej* و *.orig* التي تلقيها خلف ظهرك، لعمل مقارنة لها وإصلاحها يدويا باستخدام محرر النصوص المفضل لديك).

عملية الدمج هذه يمكن أن تكون جزءا أكثر صعوبة إذا قمت بعمل هذه التعديلات على أجزء سلسلة ملفات مصدرية، قد قام أشخاص آخرون بإجراء تعديلات عليها. فإذا قمت بعملية التطوير هذه، فأوصيك بشدة بأن تحصل على هذه المجموعة الممتازة من باتشات وبرامج (والموجودة على *http://cyberelk.net/tim/patchutils*). هذه البرامج تتيح لك التغيير في الباتشات النصية بسهولة بجميع الطرق النافعة، وتنقذ مطوري النواة من بذل ساعات من العمل الشاق.

إدارةباتشاتك بواسطة quilt

تطوير النواة باستخدام *patch* و *diff* تعمل عامة بشكل جيد. ولكن بعد برهة من الزمن، معظم الناس يبلغ درجة من التعب من البحث عن وسيلة مختلفة لعمل ذلك بشكل لا ينطوي على الكثير من الملل في الترميم والدمج. ولحسن الحظ ، جاءنا قليل من مطوري النواة ببرنامج يسمى *quilt*، يعالج عملية التلاعب بعدد من الباتشات التي صنعت من أجل سلسلة ملفات مصدرية خارجية أكثر سهولة .

وجاءت فكرةبرنامج *quilt* من مجموعة سكربتات كتبت بواسطة أندرو مورتون، حيث استخدمه في البداية لصيانة نظام إدارة الذاكرة وبعد ذلك استخدم أخيرا في تطوير الداخلي لشجرة النواة . وقد كانت سكربتاته ذات صلة وثيقة بمجال عمله ،

ولكن الأفكار وراء هذه السكربتات كانت قوية جدا. ثم قام أندرياس غروينباتشر بأخذ هذه الأفكار وانشأ الأداة *quilt*.

الفكرة الأساسية وراء *quilt* هي أنك تقوم بالعمل مع شجرة ملفات مصدرية بكر، وتضيف حفنة من الباتشات على قمتها. ويمكنك وضع أو إسقاط باتشات مختلفة لشجرة المصدر، والاحتفاظ بهذه القائمة من الباتشات بطريقة سهلة.

1. في البداية قم بإنشاء الشجرة المصدرية للنواة التي تشبها دائما :

```
$ tar -zxf linux-2.6.19.tar.gz
$ ls
linux-2.6.19/
```

2. ثم اذهب إلى هذا المجلد .

```
$ cd linux-2.6.19
```

3. لكي تبدأ العمل قم بإنشاء مجلد يسمى *patches* والذي سيحمل كل باتشات النواة الخاصة بنا.

```
$ mkdir patches
```

4. بعد ذلك أبلغ *quilt* بأن ينشئ باتشا جديدا يدعى *patch1*:

```
$ quilt new patch1
Patch patches/patch1 is now on top
```

5. يحتاج *quilt* لإبلاغه حول جميع الملفات المختلفة التي سوف يقوم بتعديلها من خلال الباتش الجديد.ولعمل ذلك ، استخدم الأمر *add* :

```
$ quilt add Makefile
File Makefile added to patch patches/patch1
```

6. قم بتحرير الملف Makefile، والتعديل على سطر EXTRAVERSION، ثم احفظ التغيرات.بعد انتهائك ، ابلغ *quilt* لتحديث الباتش :

```
$ quilt refresh
Refreshed patch patches/patch1
```

سيحتوي الملف *patches/patch1* على الباتش مع التغييرات التي قمت بعملها آنفا :

```
$ cat patches/patch1
```

```
Index: linux-2.6.19/Makefile
===================================================================
--- linux-2.6.19.orig/Makefile
+++ linux-2.6.19/Makefile
@@ -1,7 +1,7 @@
 VERSION = 2
 PATCHLEVEL = 6
```

```
 SUBLEVEL = 19
-EXTRAVERSION =
+EXTRAVERSION = -dirty
 NAME=Crazed Snow-Weasel
 # *DOCUMENTATION*
```

يمكنك مواصلة العمل قدما، مع هذا الباتش المفرد ، أو إنشاء واحد آخر جديد ليمضي فوق هذا الباتش . على سبيل المثال ، إذا تم إنشاء ثلاثة باتشات مختلفة، patch1، و patch2، و patch3، فسوف يتم تطبيقهم واحدا فوق الآخر.

لرؤية قائمة بالباتشات التي تنفذ حاليا :

```
$ quilt series -v
+ patches/patch1
+ patches/patch2
= patches/patch3
```

هذا الناتج يبين أن الثلاثة الباتشات قد تم تطبيقها، وأن الباتش الحالي هو patch3

إذا تم إطلاق إصدار جديد لنواة ، وتريد نقل التعديلات الخاصة بك إلى النسخة الجديدة، فإن *quilt* يمكنه التعامل مع ذلك بسهولة من خلال الخطوات التالية :

1. قم بإسقاط جميع الباتشات الموجودة حاليا في السلسلة :

```
$ quilt pop -a
Removing patch patches/patch3
Restoring drivers/usb/Makefile
Removing patch patches/patch2
Restoring drivers/Makefile
Removing patch patches/patch1
Restoring Makefile
No patches applied
```

2. قم باستخدام الباتش الرسمي من موقع ، *kernel.org* وقم بنقل نسخة *النواة القديمة بمقدار نسخة للأمام :*

```
$ patch -p1 < ../patch-2.6.20
$ cd ..
$ mv linux-2.6.19 linux-2.6.20
```

3. الآن اجعل *quilt* يعيد كل الباتشات الماضية على قمة الشجرة الجديدة:

```
$ quilt push
Applying patch patches/patch1
patching file Makefile
Hunk #1 FAILED at 1.
1 out of 1 hunk FAILED -- rejects in file Makefile
Patch patches/patch1 does not apply (enforce with -f)
```

4. وكما إن الباتش الأول لم يتم تبيقه بشكل سليم، افرض تنفيذ الباتش ثم بعد

ذلك قم بالترتيبات الآتية:

```
$ quilt push -f
Applying patch patches/patch1
patching file Makefile
Hunk #1 FAILED at 1.
1 out of 1 hunk FAILED -- saving rejects to
fileMakefile.rej
Aplied patch patches/patch1 (forced; needs refresh)
$ vim Makefile.rej Makefile
```

5. بعد تطبيق الباتش يدويا، قم بتحديثه :

```
$ quilt refresh
Refreshed patch patches/patch1
```

6. ثم واصل الدفع للباتشات الأخرى :

```
$ quilt push
Applying patch patches/patch2
patching file drivers/Makefile
Now at patch patches/patch2
$ quilt push
Applying patch patches/patch3
patching file drivers/usb/Makefile
Now at patch patches/patch3
```

ويوجد ايضا لدى *quilt* خيارات تستطيع أن ترسل الرسائل الإلكترونية تلقائيا بجميع الباتشات في السلسلة إلى مجموعة من الأشخاص أو القائمة البريدية ، وإلغاء باتشات معينة في وسط السلسلة ، او الذهاب لأعلى أو أسفل في سلسلة الباتشات ، حتى يجد الباتش المخصص، والمزيد من الخيارات المتعددة والمفيدة .

إذا كنت ترغب في عمل أي نوع من أشكال تطوير النواة ، فإن *quilt* يوصى به بشدة ، حتى ولو من أجل تعقب القليل من الباتشات ، بدلا من استعمال المزيد من الطرق الصعبة ل *diff* و *patch* . فهو أكثر سهولة وسوف يوفر لك الكثير من الوقت والجهد.

هنا ملاحظة شخصية ، وهي أني لا أستطيع أن أوصي بهذه الأداة بشكل كاف، حيث أني أستعملها كل يوم لإدارة مئات الباتشات لمختلف سلاسل التطوير . وهي أيضا مستخدمة من قبل العديد من توزيعات لينكس لصيانة حزم النواة الخاصة بهم ، والحصول على مجتمع للتطوير مترابط وسريع الاستجابة.

git

git هو أداة تحكم ذات شفرة مصدرية ، تم كتابتها أصلا بواسطة لينوس تورفالدز ، عندما كانت نواة لينكس تتطلع إلى نظام تحكم بشفرة مصدرية. هذا هو النظام الموزع ، الذي يختلف عن أنظمة تحكم الشفرة المصدرية التقليدية ، مثل CVS، في أنها لا يجب عليها الاتصال بخادم كي تقوم بعمل إيداع للمستودع .

git وأحد من أكثر أنظمة تحكم الشفرة المصدرية في القوة والمرنة، والسرعة؛ والمتاحة في الوقت المعاصر، ولديها طاقم تطوير نشط يقف وراءها. الصفحة الرئيسية *ل git* يمكن العثور عليها على http://git.or.cz. ومن الموصى به لكل مستخدم جديد ل *git* أن يتجول خلال المواد التعليمية المنشورة كي يعتاد على كيفية عمل *git* ، وكيفية استخدامه بشكل سليم.

يتم تطوير نواة لينكس من خلال *git*، وآخر شجرة نواة خاصة ب *git* يمكن العثور *عليها على* /http://www.kernel.org/git . إضافة إلى قائمة كبيرة من توزيعات *git* الخاصة بمطوري النواة .

استخدام *git* من الأمور الضرورية في تطوير نواة لينكس، ولكنه مناسب جدا في المساعدة على تسجيل أخطاء النواة. فإذا قمت بعمل تقرير عن خطأ ما إلى مطوري النواة ، فربما يطلبون منك استخدام *git bisect* من أجل إيجاد تغير حقيقي يتسبب في نشوء هذا الخطأ. فإذا كان ذلك ، اتبع التعليمات الموجودة في وثائق *git* لمعرفة كيفية استخدامه.

Ketchup

ketchup هو أحد الأدوات السهلة يستخدم في تحديث أو الانتقال بين النسخ المختلفة لنواة لينكس .ولديه القدرة على فعل ما يلي :

- العثور على آخر نسخة من النواة، وتحميلها ، وفك ضغطها .
- تحديث نسخة النواة الحالية المثبتة إلى أي إصدار آخر، عن طريق عمل patching للنواة إلى النسخة المناسبة .
- التعامل مع الفروع المختلفة والمستقرة من تطوير النواة، فضلا عن شجرات النواة من نوع *mm* و *stable*.
- تحميل أي باتشات أو حزم أرشيف *tarball* لازمة لعمل التحديث، إذا لم تكن موجودة على الجهاز بالفعل.
- التحقق من توقيعات *GPG* [(1)] لحزم أرشيف *tarball* والباتشات للتحقق من

GPG(1) : اختصار ل GNU Privacy Guard ،وهو معيار جنو للتشفير

أنه قام بتحميل الملف الصحيح.

يمكن العثور على ketchup على http://www.selenic.com/ketchup.

وهنا مجموعة من الخطوات توضح مدى سهولة استخدام ketchup في تحميل نسخة معينة للنواة، ثم تحويلها بعد ذلك إلى مجلد آخر لنواة لينكس ، عن طريق الحد الأدنى من الأوامر .

لجعل *ketchup* يقوم بتحميل نسخة *2.6.16.24* من النواة داخل مجلد ما، ثم إعادة تسمية المجلد ليكون بنفس اسم نسخة النواة ؛ اكتب :

```
$ mkdir foo
$ cd foo
$ ketchup -r 2.6.16.24
None -> 2.6.16.24
Unpacking linux-2.6.17.tar.bz2
Applying patch-2.6.17.bz2 -R
Applying patch-2.6.16.24.bz2
Current directory renamed to
/home/gregkh/linux/linux-2.6.16.24
```

والآن لعمل ترقية لهذه النواة لتشتمل على آخر نسخة نواة مستقرة فقط اكتب :

```
$ ketchup -r 2.6
2.6.16.24 -> 2.6.17.11
Applying patch-2.6.16.24.bz2 -R
Applying patch-2.6.17.bz2
Downloading patch-2.6.17.11.bz2
--22:21:14--
http://www.kernel.org/pub/linux/kernel/v2.6/patch-2.6
.17.11.
bz2
           =>
`/home/greg/.ketchup/patch-2.6.17.11.bz2.partial'
Resolving www.kernel.org... 204.152.191.37,
204.152.191.5
Connecting to www.kernel.org|204.152.191.37|:80...
connected.
HTTP request sent, awaiting response... 200 OK
Length: 36,809 (36K) [application/x-bzip2]
100%[=====================================>] 36,809
93.32K/s
22:21:14 (92.87 KB/s) -
`/home/greg/.ketchup/patch-2.6.17.11.bz2.partial'
saved [36809/36809]
Downloading patch-2.6.17.11.bz2.sign
--22:21:14--
```

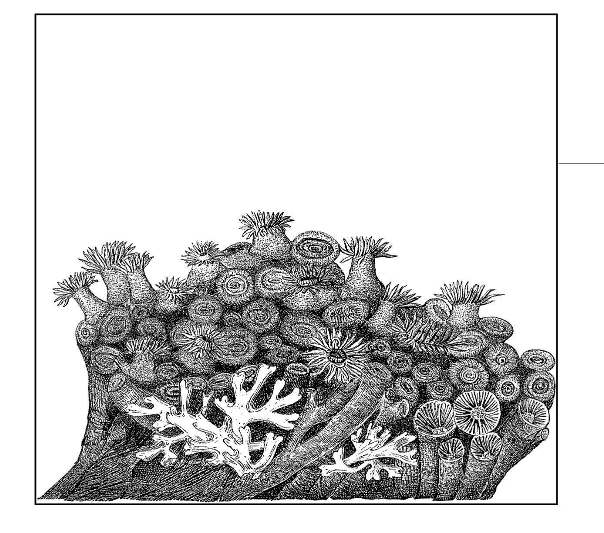
```
http://www.kernel.org/pub/linux/kernel/v2.6/patch-2.6
.17.11.
bz2.sign
           =>
`/home/greg/.ketchup/patch-2.6.17.11.bz2.sign.partial
'
Resolving www.kernel.org... 204.152.191.37,
204.152.191.5
Connecting to www.kernel.org|204.152.191.37|:80...
connected.
HTTP request sent, awaiting response... 200 OK
Length: 248 [application/pgp-signature]
100%[=====================================>] 248
--.--K/s
22:21:14 (21.50 MB/s) -
`/home/greg/.ketchup/patch-2.6.17.11.bz2.sign.
partial' saved [248/248]
Verifying signature...
gpg: Signature made Wed Aug 23 15:01:04 2006 PDT
using DSA key ID 517D0F0E
gpg: Good signature from "Linux Kernel Archives
Verification Key >
ftpadmin@kernel.org<"
gpg: WARNING: This key is not certified with a
trusted signature!
gpg:          There is no indication that the
signature belongs to the
owner.
Primary key fingerprint: C75D C40A 11D7 AF88 9981
ED5B C86B A06A 517D 0F0E
Applying patch-2.6.17.11.bz2
Current directory renamed to /home/greg/linux/tmp/x/
linux-2.6.17.11
```

وذلك يوضح لك أن ketchup قام تلقائيا بالتحقق من أن أحدث نسخة مستقرة من النواة هي *2.6.17.11*، ثم قام بتحميل ملفات الباتش اللازم للحصول على هذه النسخة من النواة.

ومن الموصى به بشدة أن تستخدم *ketchup* إذا كنت تريد تحميل أي شجرة مصدرية لنواة لينكس . حيث إنه يقوم بكل العمل للعثور على الخادم الموجود عليه الملف الصحيح للباتش ، ويقوم تلقائيا بتطبيق هذا الباتش بالصيغة المناسبة، بعد التحقق من أن الملف المحمل موقع عليه بشكل صحيح .

قم بالجمع بين *ketchup* و *quilt* وسوف تحصل على إعداد قوي يحتوي على كل شيء تحتاجه من أجل التعامل بفاعلية مع مصادر النواة وكأنك أحد مطوري النواة .

ملحق **B**



المـراجــــــــــع

أغلب المعلومات الواردة في هذا الكتاب قد تم استخلاصها من وثائق النواة والشفرة المصدرية .وهذا هو أفضل مكان للمعلومات عن كيفية بناء وتنصيب النواة ، وعادة يتم تحديثه عندما يقع أي تغيير في نظام البناء .

الكتب

هناك عدد من أجود الكتب المتاحة في برمجة نواة لينكس ، ولكن القليل منها فقط الذي يتعامل مع بناء وتثبيت النواة .
وهنا قائمة بالكتب التي وجدت أنها مفيدة عند التعامل مع نواة لينكس .

كتب لينكس العامة

* Ellen Siever, Aaron Weber, Stephen Figgins, Robert Love, and rnold Robbins. Linux in a Nutshell (O'Reilly), 2005.

هذا الكتاب أشمل وأوثق مرجع في نواة لينكس . وهو يغطي تقريبا كل أمر منفرد

سوف تحتاج إليه دوما.

- Yaghmour, Karim. Building Embedded Linux Systems (O'Reilly), 2003

هذا الكتاب ، على الرغم من انه موجه أساسا إلى مطوري لينكس ، إلا أنه يحتوي على قسم كبير بشأن كيفية بناء سلسلة أدوات المصنف المتعدي cross-compiler و بناء النواة.
وهذا الباب هو مما يوصى به للغاية، فضلا عن أجزاء أخرى من الكتاب، لأولئك الأشخاص الراغبين في معرفة المزيد عن كيفية تخصيص نواة لينكس وبقية النظام.

كتب نواة لينكس

أغلب هذه الكتب موجه أساسا إلى المبرمج الذي يهتم بتعلم كيفية البرمجة تحت بيئة لينكس . وهذه الكتب أكثر من الناحية الفنية من كتابنا هذا، ولكنه يعتبر أعظم مكان يمكنك الانطلاق منه إذا كنت ترغب في تعلم المزيد عن الشفرة التي تتحكم في لينكس.

- Jonathan Corbet, Alessandro Rubini, and Greg Kroah-Hartman. Linux Device Drivers (O'Reilly), 2005.

هذا الكتاب يغطي كيفية عمل النظم الفرعية لمشغلات نواة لينكس المختلفة ، ويقدم العديد من الأمثلة عن المشغلات العاملة . ويوصى به لاي شخص يريد العمل على مشغلات نواة لينكس . وهو متاح أيضا مباشرة على الشبكة للتداول مجانا على : http://lwn.net/Kernel/LDD3/ .

- Love, Robert. Linux Kernel Development (Novell Press Publishing), 2005.

كتاب روبرت لاف يغطي معظم النواحي الخاصة بنواة لينكس ، ويوضح كيفية عمل كل شيء كذلك. وهو أعظ مكان يمكن الانطلاق منه لتعلم أجزاء مختلفة عن دواخل نواة لينكس .

- Bovet, Daniel P. and Cesate, Marco. Understanding

the Linux Kernel (O'Reilly), 2005.

هذا الكتاب يبحث في تصميم وتنفيذ قلب نواة لينكس.وهو مرجع كبير لفهم الخوارزميات المستخدمة في مختلف أجزاء من النواة. وهو مما يوصى به للغاية لأي شخص يريد فهم التفاصيل عن كيفية عمل النواة.

أماكن الأدوات

هناك الكثير من الأدوات الاختلاف المذكورة في هذا الكتاب. وهنا روابط تشير إلى حيث توجد شفرة المصدر لهذه الأدوات على شبكة الانترنت .

Linux kernel

http://www.kernel.org و *ftp://ftp.kernel.org* يحتويان على جميع النسخ المختلفة لشفرة الملف المصدري لنواة لينكس .

http://www.kernel.org/git ويحتوي على قائمة لكل هياكل *git* المستخدمة من قبل مختلف مطوري نواة لينكس.

Gcc

http://gcc.gnu.org/ هو الموقع الرئيسي لكل شي مرتبط بمصنف *GNU C* .

binutils

http://www.gnu.org/software/binutils/ هو الموقع الرئيسي لكل المعلومات حول الأداة *binutils*.

Make

http://www.gnu.org/software/make/ هو الموقع الرئيسي لكل المعلومات حول *Make* .

Util-linux

http://www.kernel.org/pub/linux/utils/util-linux هو الدليل الذي يمكنك منه تحميل جميع النسخ من *util-linux* .

Module-init-tools

http://www.kernel.org/pub/linux/utils/kernel/module-init-tools هو

الدليل الذي يمكنك منه تحميل جميع النسخ من *module-init-tools* .

E2fsprogs

http://e2fsprogs.sourceforge.net هي الصفحة الرئيسية لمشروع الحزمة *e2fsprogs* .

Jfsutils

http://jfs.sourceforge.net هي الصفحة الرئيسية لمشروع الحزمة *Jfsutils*

reiserfsprogs

http://www.namesys.com/download.html هي الصفحة الرئيسية لمشروع الحزمة *reiserfsprogs* .

Xfsprogs

http://oss.sgi.com/projects/xfs هي الصفحة الرئيسية لمشروع الحزمة *Xfsprogs*

quota-tools

http://sourceforge.net/projects/linuxquota/ هي الصفحة الرئيسية لمشروع الحزمة *quota-tools*

nfs-utils

http://nfs.sf.net/ هي الصفحة الرئيسية لمشروع *nfs-utils* .

Udev

http://www.kernel.org/pub/linux/utils/kernel/hotplug/udev.html . هو الدليل الذي يمكن من خلاله تحميل جميع النسخ من *udev* .

Procfs

http://procps.sourceforge.net/ هي الصفحة الرئيسية لمشروع حزمة *Procfs* .

git

http://git.or.cz/ هي الصفحة الرئيسية لمشروع *git* .

ketchup

http://www.selenic.com/ketchup/ هي الصفحة الرئيسية لمشروع برنامج *ketchup*

quilt

http://savannah.nongnu.org/projects/quilt هي الصفحة الرئيسية لمشروع برنامج *quilt* .

distcc

http://distcc.samba.org/ هي الصفحة الرئيسية لمشروع برنامج *distcc* .

ccache

http://ccache.samba.org هي الصفحة الرئيسية لمشروع لبرنامج *ccache*.